PERVASIVE SOFTWARE INC. LICENSES THE SOFTWARE AND DOCUMENTATION PRODUCT TO YOU OR YOUR COMPANY SOLELY ON AN “AS IS” BASIS AND SOLELY IN ACCORDANCE WITH THE TERMS AND CONDITIONS OF THE ACCOMPANYING LICENSE AGREEMENT. PERVASIVE SOFTWARE INC. MAKES NO OTHER WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, REGARDING THE SOFTWARE OR THE CONTENT OF THE DOCUMENTATION; PERVASIVE SOFTWARE INC. HEREBY EXPRESSLY STATES AND YOU OR YOUR COMPANY ACKNOWLEDGES THAT PERVASIVE SOFTWARE INC. DOES NOT MAKE ANY WARRANTIES, INCLUDING, FOR EXAMPLE, WITH RESPECT TO MERCHANTABILITY, TITLE, OR FITNESS FOR ANY PARTICULAR PURPOSE OR ARISING FROM COURSE OF DEALING OR USAGE OF TRADE, AMONG OTHERS.

Btrieve, Client/Server in a Box, Pervasive, Pervasive Software, and the Pervasive Software logo are registered trademarks of Pervasive Software Inc.
Built on Pervasive Software,DataExchange, MicroKernel Database Engine, MicroKernel Database Architecture, Pervasive.SQL, Pervasive PSQL, Solution Network, Ultralight, and ZDBA are trademarks of Pervasive Software Inc.
NetWare and Novell are registered trademarks of Novell, Inc.
NetWare Loadable Module, NLM, Novell DOS, Transaction Tracking System, and TTS are trademarks of Novell, Inc.
Sun, Sun Microsystems, Java, all trademarks and logos that contain Sun, Solaris, or Java, are trademarks or registered trademarks of Sun Microsystems.
All other company and product names are the trademarks or registered trademarks of their respective companies.
© Copyright 2013 Pervasive Software Inc. All rights reserved. Reproduction, photocopying, or transmittal of this publication, or portions of this publication, is prohibited without the express prior written consent of the publisher.
This product includes software developed by Powerdog Industries. © Copyright 1994 Powerdog Industries. All rights reserved.
This product includes software developed by KeyWorks Software. © Copyright 2002 KeyWorks Software. All rights reserved.
This product includes software developed by DUNDAS SOFTWARE. © Copyright 1997-2000 DUNDAS SOFTWARE LTD., all rights reserved.
This product includes software developed by the Apache Software Foundation (http://www.apache.org/).
This product uses thefree unixODBC Driver Manager as written by Peter Harvey (pharvey@codebydesign.com), modified and extended by Nick Gorham (nick@easysoft.com), with local modifications from Pervasive Software. Pervasive Software will donate their code changes to the current maintainer of the unixODBC Driver Manager project, in accordance with the LGPL license agreement of this project. The unixODBC Driver Manager home page is located at www.unixodbc.org. For further information on this project, contact its current maintainer: Nick Gorham (nick@easysoft.com).
A copy of the GNU Lesser General Public License (LGPL) is included on the distribution media for this product. You may also view the LGPL at www.fsf.org/licensing/licenses/lgpl.html.

User’s Guide
January 2013
138-004434-004
Contents

About This Manual ................................................................. xiii
Who Should Read This Manual ........................................... xiv
Manual Organization ............................................................ xv
Conventions ........................................................................... xvii

1 Introducing Pervasive PSQL .................................................. 1
Understanding Pervasive PSQL and its Capabilities
Understanding the Pervasive PSQL Database Management System ........................................... 2
What is a Database? ................................................................. 2
What is a Database Management System? ..................................... 3
Components of Pervasive PSQL .................................................. 3
Key Concepts .......................................................................... 7
Basic Database Structures and Terms .......................................... 7
Unique Benefits of Pervasive PSQL .............................................. 10
Understanding the DBMS Products ............................................. 12
Pervasive PSQL Workgroup ....................................................... 12
Pervasive PSQL Server .............................................................. 13
Pervasive PSQL Vx Server ......................................................... 13
Crystal Reports for Pervasive PSQL .............................................. 13
Using Pervasive PSQL Documentation ........................................ 14
Getting Started ....................................................................... 14
What Is New ........................................................................... 14
User’s Guide ........................................................................... 15
Advanced Operations Guide ..................................................... 15
SQL Engine Reference .............................................................. 15
Status Codes and Messages ....................................................... 15
Pervasive PSQL Vx Product Guide .............................................. 15
Additional Information ............................................................. 16
File System Security ................................................................. 17

2 Using Pervasive PSQL ............................................................ 19
A Walk-through of Basic User Tasks
Starting and Stopping the Database Engine ................................ 20
Starting and Stopping the Server Engine on a Windows Server .... 20
Starting and Stopping the Workgroup Engine on Windows ........ 23
Starting and Stopping the Database Engine on Linux ................. 25
Granting Administrative Rights for the Database Engine ............. 26
Tasks Requiring Administrative Rights ...................................... 26
How Administrative Rights are Granted .................................... 26
Rights Within an Active Directory Environment ......................... 27
## Contents

Rights Provided to non-Administrative Users .............................................. 27
Tasks for Granting Administrative Rights ................................................. 27
Granting Administrative Rights on a Windows Server ................................. 27
Granting Administrator Rights on Linux .................................................... 31
Logging in as Administrator on any platform ............................................ 32
Setting Up ODBC Database Access ............................................................. 33
ODBC Standard ............................................................................................. 33
Servers and Clients ....................................................................................... 34
Data Source Names ....................................................................................... 35
Internal Database Name ................................................................................ 35
Applications Using the Transactional Interface ............................................. 35
Setting up Database Access with PCC ......................................................... 36
Setting Up Database Access on Windows ...................................................... 36
Setting Up Database Access on a Linux Server ............................................ 37
Setting Up Client Access from a Windows Client ........................................ 39
Setting Up a Client DSN on a Linux Client .................................................. 41
Accessing Data on a Remote Engine Using PCC .......................................... 42
Accessing Data via ODBC From Other Applications .................................... 45
Before You Begin ......................................................................................... 45
Accessing Data Using Microsoft Excel ......................................................... 45
Accessing Data Using Microsoft Access ...................................................... 47
Deleting DSNs ............................................................................................... 51
Using the Fast User Switching Feature of Windows XP ................................. 53

### 3 Using Pervasive PSQL Control Center ..................................................... 55

A Tour of Pervasive PSQL Control Center

An Overview of Pervasive PSQL Control Center ............................................ 56
Installing PCC ............................................................................................... 57
Starting PCC on Windows ............................................................................. 57
Starting PCC On Linux .................................................................................. 58
Situations Requiring That You Clear PCC Cache ........................................ 60
Editors and Views Within PCC ..................................................................... 61
Preferences ................................................................................................. 64
Additional Utilities ...................................................................................... 66
External Tools ............................................................................................... 67
Services on Windows Servers ....................................................................... 68
Servers Properties ....................................................................................... 69
Database Engines ....................................................................................... 70
Database Engine Properties ....................................................................... 71
Capacity Usage Viewer ............................................................................... 72
Capacity Usage Viewer GUI ........................................................................ 72
Zooming ....................................................................................................... 75
Monitor ........................................................................................................ 77
Accessing Monitor ...................................................................................... 77
Features of the Monitor User Interface ....................................................... 77
### Contents

Overview ............................................. 162
Table Editor Pages .................................. 162
Table Editor Graphical User Interface .......... 166
Columns Page ....................................... 166
Indexes Page ....................................... 168
Foreign Keys Page .................................. 168
SQL View Page ...................................... 168
Table Editor Tasks .................................. 169
General Tasks ...................................... 169
Columns Tasks ...................................... 170
Indexes Tasks ...................................... 171
Foreign Keys Tasks .................................. 171
SQL View Tasks ..................................... 171

6 SQL Editor ......................................... 201
A Tour of the SQL Editor
SQL Editor Concepts ................................ 202
Overview ............................................ 202
Restrictions ........................................ 203
Displaying Statement Results ..................... 203
Outline Window View ................................ 207
Working with Common SQL Objects ............... 208
SQL Editor Used in SQL View Tab of Table Editor 210
SQL Editor Tasks .................................... 211
General Tasks ...................................... 211
Execution Tasks .................................... 212
Grid Tasks .......................................... 212
Text View Tasks .................................... 212
Outline View Tasks .................................. 212
Common SQL Object Tasks ......................... 212

7 Pervasive System Analyzer (PSA) ............... 229
Usage Information for the Diagnostic Utility PSA
PSA Concepts ........................................ 230
Frequently Asked Questions ....................... 232
PSA GUI Visual Reference ......................... 234
PSA Tasks .......................................... 236
General Tasks ...................................... 236
View Modules Tasks ................................ 236
Test Active Installation Tasks ................... 238
Log Files Tasks ..................................... 241

8 Command Line Interface Utilities ............... 243
Using Pervasive PSQL Command Line Interface Utilities
CLI Utilities Overview ................................................................. 244
Platforms that Include CLI Utilities ........................................... 244
Where to Find CLI Utilities ........................................................ 244
Utilities by Platform and Engine Type ........................................ 244
Command Line Interface Utility Reference ................................. 248
bcfg ............................................................. 249
bdu ............................................................. 250
Description ........................................................................... 250
Synopsis ................................................................................. 251
Options ................................................................................. 251
Notes .................................................................................. 252
Examples .............................................................................. 254
bmon ................................................................. 257
btadmin .............................................................. 258
Description ........................................................................... 258
Synopsis ................................................................................. 258
Options ................................................................................. 258
See Also .............................................................................. 258
Notes .................................................................................. 258
butil ................................................................. 260
Description ........................................................................... 260
Synopsis ................................................................................. 260
Options ................................................................................. 261
See Also .............................................................................. 261
Notes .................................................................................. 258
clicaladm .......................................................... 262
Description ........................................................................... 262
Synopsis ................................................................................. 262
Options ................................................................................. 262
See Also .............................................................................. 262
Notes .................................................................................. 258
clipaadm .......................................................... 263
Description ........................................................................... 263
Synopsis ................................................................................. 263
Options ................................................................................. 263
See Also .............................................................................. 263
Notes .................................................................................. 263
dbmaint .............................................................. 264
Description ........................................................................... 264
Synopsis ................................................................................. 264
Options ................................................................................. 264
See Also .............................................................................. 266
Notes .................................................................................. 270
dsnadd .............................................................. 267
Description ........................................................................... 267
Synopsis ................................................................................. 267
Options ................................................................................. 268
Examples .............................................................................. 269
Notes .................................................................................. 270
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Also</td>
<td>273</td>
</tr>
<tr>
<td>isid</td>
<td>274</td>
</tr>
<tr>
<td>Description</td>
<td>274</td>
</tr>
<tr>
<td>Synopsis</td>
<td>275</td>
</tr>
<tr>
<td>Options</td>
<td>275</td>
</tr>
<tr>
<td>Commands</td>
<td>275</td>
</tr>
<tr>
<td>Notes</td>
<td>276</td>
</tr>
<tr>
<td>Examples</td>
<td>276</td>
</tr>
<tr>
<td>See Also</td>
<td>277</td>
</tr>
<tr>
<td>licgetauth</td>
<td>278</td>
</tr>
<tr>
<td>Description</td>
<td>278</td>
</tr>
<tr>
<td>Synopsis</td>
<td>278</td>
</tr>
<tr>
<td>Options</td>
<td>278</td>
</tr>
<tr>
<td>See Also</td>
<td>278</td>
</tr>
<tr>
<td>ps .</td>
<td>279</td>
</tr>
<tr>
<td>Description</td>
<td>279</td>
</tr>
<tr>
<td>Synopsis</td>
<td>279</td>
</tr>
<tr>
<td>Options</td>
<td>279</td>
</tr>
<tr>
<td>Return Codes</td>
<td>280</td>
</tr>
<tr>
<td>psregedit</td>
<td>282</td>
</tr>
<tr>
<td>Description</td>
<td>282</td>
</tr>
<tr>
<td>Synopsis</td>
<td>282</td>
</tr>
<tr>
<td>Options</td>
<td>282</td>
</tr>
<tr>
<td>psregsvr</td>
<td>284</td>
</tr>
<tr>
<td>Description</td>
<td>284</td>
</tr>
<tr>
<td>Synopsis</td>
<td>284</td>
</tr>
<tr>
<td>Options</td>
<td>284</td>
</tr>
<tr>
<td>pvdbpass</td>
<td>285</td>
</tr>
<tr>
<td>Description</td>
<td>285</td>
</tr>
<tr>
<td>Synopsis</td>
<td>285</td>
</tr>
<tr>
<td>Options</td>
<td>285</td>
</tr>
<tr>
<td>Examples</td>
<td>285</td>
</tr>
<tr>
<td>pvddl</td>
<td>287</td>
</tr>
<tr>
<td>Description</td>
<td>287</td>
</tr>
<tr>
<td>Synopsis</td>
<td>287</td>
</tr>
<tr>
<td>Options</td>
<td>287</td>
</tr>
<tr>
<td>See Also</td>
<td>288</td>
</tr>
<tr>
<td>pvmconv</td>
<td>289</td>
</tr>
<tr>
<td>Synopsis</td>
<td>289</td>
</tr>
<tr>
<td>Parameters</td>
<td>289</td>
</tr>
<tr>
<td>Description</td>
<td>290</td>
</tr>
<tr>
<td>Conversion From V1 to V2 Metadata</td>
<td>291</td>
</tr>
<tr>
<td>Conversion From V2 to V1 Metadata</td>
<td>291</td>
</tr>
<tr>
<td>Examples</td>
<td>291</td>
</tr>
<tr>
<td>pvmnetpass</td>
<td>294</td>
</tr>
</tbody>
</table>
Contents

Description ................................................................. 294
Synopsis ........................................................................ 294
Options .......................................................................... 294
See Also ........................................................................ 295
Examples ......................................................................... 295
rbldcli .......................................................................... 297
Description .................................................................... 297
Synopsis ........................................................................ 297
Options .......................................................................... 297
See Also ........................................................................ 297

9 Basic Troubleshooting ............................................... 299
How to Identify and Solve Common Problems
Pervasive PSQL Message Logging ................................. 300
Licensing Messages .......................................................... 300
Pervasive Notification Viewer ........................................... 302
Operating System Event Log ............................................ 305
Pervasive PSQL Event Log ............................................... 307
General Troubleshooting ................................................. 310
Error Messages from PCC ................................................ 316
Frequently Asked Questions ............................................ 320
Installation ..................................................................... 325
PCC ............................................................................ 326
Security .......................................................................... 326
User Counts .................................................................... 327
Networking ..................................................................... 329
Difficulty Accessing Data ............................................... 329
ODBC and DDFs ............................................................. 331
Upgrading from Btrieve 6.15 ............................................ 334
Upgrading and Migration ............................................... 335
Miscellaneous ............................................................... 335

10 Pervasive PSQL Resources and Contacts ................. 339
A Guide to Pervasive PSQL Customer Information Resources
Thirty-Day Free Technical Support .................................. 340
# Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Services Dialog Box</td>
<td>22</td>
</tr>
<tr>
<td>2</td>
<td>Services Dialog Box</td>
<td>23</td>
</tr>
<tr>
<td>3</td>
<td>Connect to Remote Server Dialog Box</td>
<td>32</td>
</tr>
<tr>
<td>4</td>
<td>Example DSN Configurations</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>Pervasive ODBC Interface for Client DSNs</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td>Advanced Connection Attributes for Client DSNs</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>Expanding the Databases List for an Engine</td>
<td>42</td>
</tr>
<tr>
<td>8</td>
<td>Selecting the Dept Table in DEMODATA</td>
<td>43</td>
</tr>
<tr>
<td>9</td>
<td>Displaying the Dept Table in DEMODATA</td>
<td>43</td>
</tr>
<tr>
<td>10</td>
<td>Refining Your Query - Dept Table in DEMODATA</td>
<td>44</td>
</tr>
<tr>
<td>11</td>
<td>Accessing Pervasive Data using Microsoft Excel</td>
<td>46</td>
</tr>
<tr>
<td>12</td>
<td>Excel Display of ODBC Source List</td>
<td>46</td>
</tr>
<tr>
<td>13</td>
<td>Create a New Database using Microsoft Access</td>
<td>47</td>
</tr>
<tr>
<td>14</td>
<td>Importing External Data Using Access</td>
<td>48</td>
</tr>
<tr>
<td>15</td>
<td>Access Display of ODBC Source List</td>
<td>49</td>
</tr>
<tr>
<td>16</td>
<td>Using Pervasive Data in Microsoft Access</td>
<td>50</td>
</tr>
<tr>
<td>17</td>
<td>Pervasive PSQL Control Center on Windows Platforms</td>
<td>56</td>
</tr>
<tr>
<td>18</td>
<td>Pervasive PSQL Control Center on Linux Platforms</td>
<td>57</td>
</tr>
<tr>
<td>19</td>
<td>Example Objects Shown in Pervasive PSQL Explorer</td>
<td>63</td>
</tr>
<tr>
<td>20</td>
<td>Capacity Usage Viewer User Interface</td>
<td>73</td>
</tr>
<tr>
<td>21</td>
<td>Statistical Indicators</td>
<td>74</td>
</tr>
<tr>
<td>22</td>
<td>Monitor User Interface</td>
<td>78</td>
</tr>
<tr>
<td>23</td>
<td>Create New Database Dialog</td>
<td>95</td>
</tr>
<tr>
<td>24</td>
<td>Columns Page</td>
<td>166</td>
</tr>
<tr>
<td>25</td>
<td>SQL Editor</td>
<td>202</td>
</tr>
<tr>
<td>26</td>
<td>Grid Window View</td>
<td>204</td>
</tr>
<tr>
<td>27</td>
<td>Text Window View</td>
<td>206</td>
</tr>
<tr>
<td>28</td>
<td>Outline Window View</td>
<td>207</td>
</tr>
<tr>
<td>29</td>
<td>PSA Main Dialog</td>
<td>234</td>
</tr>
<tr>
<td>30</td>
<td>View Modules Section of PSA</td>
<td>237</td>
</tr>
<tr>
<td>31</td>
<td>Notification Viewer GUI</td>
<td>305</td>
</tr>
</tbody>
</table>
## Tables

1. Summary of Pervasive PSQL Utilities .......................... 4
2. Example Table for Telephone Directory .......................... 7
3. Example Table for Food Preferences ............................. 9
4. Action of Dependent Services Based on Action of Database Engine Services .................. 21
5. Requirements for Starting PCC on Linux .......................... 58
6. Troubleshooting Guide for Running PCC ........................... 59
7. Characteristics of PCC Window Views ............................. 61
8. Tab Icons ............................................................... 80
9. Create New Database GUI Elements ................................. 95
10. Table Properties on General Tab .................................... 101
11. Restrictions Pertaining to Windows Platforms ..................... 135
12. Restrictions Pertaining to Linux Platforms ......................... 135
13. License Administrator Command Line Options and Parameters .......... 142
14. Description of Common SQL Objects in PCC ......................... 209
15. Delimiters for Columns .................................................. 250
16. Delimiters for Rows ....................................................... 250
17. Bulk Data Utility Parameters ............................................. 251
18. Pervasive PSQL ODBC Driver Descriptions for Linux ................. 271
19. FAQs About ODBC and DSN Support for Linux ....................... 272
20. Pervasive PSQL Message Logging Repositories ...................... 300
21. Message Type Based on Change in State of Key ...................... 301
22. Logging Frequency of Messages By Initiating Action .................. 302
23. PVSW.LOG Fields ......................................................... 308
24. Pervasive Software Resources ........................................... 340
Tables
About This Manual

This manual introduces you to Pervasive PSQL utilities for Server and Workgroup products and shows you how to perform the basic tasks necessary to work with the application. Topics include starting and stopping the database engine, setting up access to a database, and accessing data from other applications. This manual also gives you a tour of the Pervasive PSQL Control Center (PCC). PCC allows you to manage Pervasive PSQL utilities within a single, easy-to-use framework.
Who Should Read This Manual

This manual provides information for users who install and run Pervasive PSQL client/server and Workgroup products.

Pervasive Software would appreciate your comments and suggestions about this manual. As a user of our documentation, you are in a unique position to provide ideas that can have a direct impact on future releases of this and other manuals. If you have comments or suggestions for the product documentation, post your request at the Community Forum on the Pervasive Software Web site.
Manual Organization

This manual is divided into the following chapters:

- Chapter 1—Introducing Pervasive PSQL
  This chapter provides an introduction to Pervasive PSQL, and an overview of utilities and Pervasive PSQL documentation.

- Chapter 2—Using Pervasive PSQL
  This chapter covers the basic tasks required to work with Pervasive PSQL.

- Chapter 3—Using Pervasive PSQL Control Center
  This chapter explains how to get your work done using the Pervasive PSQL Control Center utility.

- Chapter 4—License Administration
  This chapter explains how to authorize and deauthorize keys, increase user count, session count limit, or data in use limit, and view license keys.

- Chapter 5—Table Editor
  This chapter explains how to work with tables and columns using Table Editor.

- Chapter 6—SQL Editor
  This chapter explains how to use SQL Editor to run SQL statements against a Pervasive PSQL database.

- Chapter 7—Pervasive System Analyzer (PSA)
  This chapter explains how to archive, restore, and remove Pervasive PSQL components and how to test network connections and active installations.

- Chapter 8—Command Line Interface Utilities
  This chapter explains the utilities that provide a command line interface.

- Chapter 9—Basic Troubleshooting
  This chapter provides information for troubleshooting and resolving problems.
Chapter 10—Pervasive PSQL Resources and Contacts

This chapter explains the resources and information at your disposal as a valued customer of Pervasive Software.

The manual also includes an index.
Conventions

Unless otherwise noted, command syntax, code, and examples use the following conventions:

**CASE**

Commands and reserved words typically appear in uppercase letters. Unless you are working with Linux or the manual states otherwise, you can enter these items using uppercase, lowercase, or both. For example, you can type MYPROG, myprog, or MYprog.

**Bold**

Words appearing in bold include the following: menu names, dialog box names, commands, options, buttons, statements, and so forth.

**Monospaced font**

Monospaced font is reserved for words you enter, such as command syntax.

**[ ]**

Square brackets enclose optional information, as in [log_name]. If information is not enclosed in square brackets, it is required.

**|**

A vertical bar indicates a choice of information to enter, as in [file name | @file name].

**< >**

Angle brackets enclose multiple choices for a required item, as in /D=<5|6|7>.

**variable**

Words appearing in italics are variables that you must replace with appropriate values, as in file name.

**...**

An ellipsis following information indicates you can repeat the information more than one time, as in [parameter ...].

**::=**

The symbol ::= means one item is defined in terms of another. For example, a::=b means the item a is defined in terms of b.
Introducing Pervasive PSQL

Understanding Pervasive PSQL and its Capabilities

This chapter provides an explanation of what Pervasive PSQL is and what it can do for you. This chapter is divided into the following sections:

- Understanding the Pervasive PSQL Database Management System
- Key Concepts
- Understanding the DBMS Products
- Using Pervasive PSQL Documentation
- File System Security
Understanding the Pervasive PSQL Database Management System

Pervasive PSQL is a comprehensive database management system built around Pervasive Software’s MicroKernel Database Engine. Pervasive PSQL offers easy installation, uncomplicated maintenance, and high levels of performance and reliability.

This section explains the product and its components.

What is a Database?

Loosely defined, a database is simply a collection of data. Generally, the data is structured by dividing it into sub-sets of information that share the same characteristics. Some examples of a database are:

- A telephone book
  Each entry in the phone book consists of four characteristics: first name, last name, address, and phone number.

- A collection of digital photographs
  Each picture on your hard disk has two characteristics: a file name, and the data within the file that represents the image.

- A list of orchards and the fruit grown by each
  Each entry in the orchard list might consist of three characteristics: orchard name, address, and date founded. The related list of fruits might have five characteristics: orchard name, fruit name, fruit type (McIntosh, Fuji, and so on), fruit price, and a taste rating.

In the particular context of this product, a database is a specific, well-defined collection of related information. You can probably find one or more databases available on your computer or your network. For example, you may have a database of information related to vendors from whom you purchase supplies or raw materials, and you probably also have a database containing customer or member information. Each of these is a distinct, well-defined collection of related information.
As citizens of the computer age, we are surrounded by collections of information—databases—everywhere we go. Unfortunately, all this data is of no use to anyone without methods to sort it, search it, analyze it, and keep it current.

A database management system, or DBMS, is a computer program designed to manage large amounts of data and to allow other computer programs and people to interact with the data. A DBMS can also be referred to informally as a database engine or simply an engine. A DBMS performs the following tasks:

- Controls access to the data. The DBMS can act as a watchdog to prevent the wrong people from using the data.
- Structures the data so it can be interpreted by other applications. The DBMS ensures that all the data adheres to the database structure, so that other computer programs can work with the data using common methods.
- Keeps the data safe and prevents it from getting garbled or lost. The DBMS facilitates backing up the data in case of catastrophic loss, and also accesses it in a consistent manner to prevent the data from suffering inadvertent damage.
- Makes it easy to add new information, find it, update it, and delete it. The DBMS readily accepts new data and provides tools that you can use to locate, update, and remove information as you see fit. It verifies that the data inserted fits within defined attributes for the database fields.
- Allows you to analyze relationships among different sets of data. The DBMS stores the data in a way that allows you to examine how any piece of data relates to any other piece of data.

In summary, the DBMS organizes your data, keeps it safe, and helps you to use it and understand it.

The Pervasive PSQL DBMS consists of a variety of components designed to help you achieve your data management goals.

**MicroKernel Database Engine**

The MicroKernel Database Engine (MKDE) is the high-performance heart of Pervasive PSQL. The MKDE works directly with the data files on your computer's hard disk. When requested, it directly inserts new data, deletes unnecessary data, and ensures the safety and
Introducing Pervasive PSQL

Integrity of the data files at all times, even when people and applications are working with the data.

**SQL Relational Database Engine**

The SQL Relational Database Engine (SRDE) interacts with the MKDE and the client (described below). It provides many powerful features including support for Microsoft ODBC, sophisticated search and analysis capability, and security.

**Client (also called Requester)**

In client/server systems, the client resides on the computer workstation. The client interacts with the client application and across the network with both the MKDE and the SRDE on the server.

**Pervasive PSQL Control Center**

The Pervasive PSQL Control Center (PCC) is an easy-to-use, graphical tool designed to help you create and manipulate databases and control your DBMS. It allows you to access nearly all the functions of the product from one place. For a tour of PCC, see Chapter 3, Using Pervasive PSQL Control Center.

**Utilities**

The Pervasive PSQL database engines come with a variety of graphical and command-line tools designed to provide support for testing, configuring, and manipulating the many features and options provided by Pervasive PSQL. Most of the utilities run on Windows and allow remote function to Linux database server engines.

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Supported platforms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pervasive PSQL Control Center</td>
<td>Windows and Linux</td>
<td>Primary utility for Pervasive PSQL. Lists engines and databases available and allows you to set properties (configure) objects. See Using Pervasive PSQL Control Center.</td>
</tr>
</tbody>
</table>
### Table 1  Summary of Pervasive PSQL Utilities  continued

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Supported platforms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function Executor</td>
<td>Windows</td>
<td>Executes Btrieve operations, enabling you to learn how the Btrieve interface works or test and debug an application. Testing Btrieve Operations in Advanced Operations Guide.</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Windows and Linux</td>
<td>Performs common Pervasive PSQL file and data manipulations, such as importing and exporting data. BUTIL is the command line version. See Manipulating Btrieve Data Files with Maintenance in Advanced Operations Guide.</td>
</tr>
<tr>
<td>SQL Editor - invoked within PCC</td>
<td>Windows and Linux</td>
<td>Allows you to execute SQL statements interactively and add or edit data in tables. See SQL Editor.</td>
</tr>
<tr>
<td>Table Editor - invoked within PCC</td>
<td>Windows and Linux</td>
<td>Allows you to add, delete, or change the characteristics of columns within a table, and to create a table. See Table Editor.</td>
</tr>
<tr>
<td>Rebuild</td>
<td>Windows</td>
<td>Converts one version of MicroKernel files into another version. See Converting Data Files in Advanced Operations Guide.</td>
</tr>
<tr>
<td>License Administrator</td>
<td>Windows and Linux</td>
<td>Manages Pervasive PSQL license keys. See License Administration.</td>
</tr>
<tr>
<td>ODBC Administrator</td>
<td>Windows</td>
<td>Sets up Data Source Names (DSNs) for client and engine interfaces. See DSNs and ODBC Administrator in SQL Engine Reference.</td>
</tr>
<tr>
<td>Gateway Locator</td>
<td>Windows</td>
<td>Used to configure and maintain gateway locator files for the Workgroup engine. See Setting Up a Gateway Configuration in Getting Started With Pervasive PSQL.</td>
</tr>
<tr>
<td>Pervasive System Analyzer</td>
<td>Windows</td>
<td>Analyzes system components and runs communication tests. See Pervasive System Analyzer (PSA).</td>
</tr>
<tr>
<td>Query Plan Viewer</td>
<td>Windows</td>
<td>Displays query plans selected by the database engine so you can better determine how to optimize SQL queries. See Query Plan Viewer in SQL Engine Reference.</td>
</tr>
</tbody>
</table>
Introducing Pervasive PSQL

Table 1  Summary of Pervasive PSQL Utilities  continued

<table>
<thead>
<tr>
<th>Utility name</th>
<th>Supported platforms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDF Builder</td>
<td>Windows</td>
<td>Allows you to view, create, and change Pervasive PSQL data dictionary files (DDFs) without modifying the underlying data file. See Getting Started with DDF Builder in DDF Builder User’s Guide.</td>
</tr>
<tr>
<td>Notification Viewer</td>
<td>Windows and Linux</td>
<td>Displays messages logged by the licensing components. Notification Viewer provides two interfaces: system tray icons and a graphical user interface (GUI). See Pervasive Notification Viewer.</td>
</tr>
</tbody>
</table>

Documentation

Pervasive PSQL comes with a complete set of online documentation. For more information about the documentation, see Using Pervasive PSQL Documentation.
Key Concepts

This section explains some basic concepts of databases and some of the key concepts that distinguish Pervasive PSQL from other database products.

Basic Database Structures and Terms

Most database management systems in use today share a common set of basic structures. This section briefly explains those structures. The descriptions that follow refer to the diagram below:

Table 2 Example Table for Telephone Directory

<table>
<thead>
<tr>
<th>Col Names</th>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 1</td>
<td>Fred Black</td>
<td>643 Oak</td>
<td>12346</td>
<td>555-2345</td>
</tr>
<tr>
<td>Row 2</td>
<td>Jane Doe</td>
<td>112 Elm</td>
<td>12345</td>
<td>555-1212</td>
</tr>
<tr>
<td>Row 3</td>
<td>John Doe</td>
<td>112 Elm</td>
<td>12345</td>
<td>555-1212</td>
</tr>
</tbody>
</table>

Value

The most basic element of a database is a value. A value is one piece of data, one characteristic, for a specific entity. For example, in the diagram, the name “John Doe” or the phone number “555-1212” is a value.

Column or Field

Another element is a column, or a field. A column represents a characteristic with no specific value. Columns generally have names that describe the given characteristic. For example, in the telephone book, Name and Phone are columns. They do not have specific values unless you look up a particular person. Field is sometimes used to refer to the generic characteristic of a specific row. For example, someone might point to a specific box in the table above and ask, “What is the value of that field?”
Introducing Pervasive PSQL

Row or Record
Another element is called a row, or a record. A row is a collection of all the values for one particular instance. For example, one entry in the phone book, complete with name, address, and phone number, is one record or row.

Cell
A cell is a column within a specific record. You can think of it as the intersection of a row and a column. Each cell has a specific value. For example, you might tell a co-worker, “The value of the cell located at row 2, column 3 is ‘12345’.”

Table
A collection of rows and columns makes up a table. A table is a set of data that shares exactly the same structure. Tables generally have names that describe the contents of the table. For example, the table above is called “Phone Book.” With Pervasive PSQL, each table is stored as a separate data file on the hard disk.

Index
An index is an ordered list of all the values in a particular column. A table can have zero or more indexes on it. The database engine uses indexes to find specific records in the database without having to step through every record one at a time. Creating indexes on columns which will frequently be used in database searches is likely to improve the performance of your database.

Database
A database is a collection of one or more tables. The data in the tables does not need to be related among the various tables, but usually there are many relations. For example, a database might consist of the “Food Preferences” table below, and the “Phone Book” table above. With Pervasive PSQL, a database consists of one or more data files and Data Dictionary Files (DDFs) on your hard disk. The DDFs are special data files that contain all the definitions for tables, columns, and other attributes that define the structure of your database.
**Key Concepts**

**Schema**

The term schema refers to the complete set of definitions that describe the entire structure of a database. A typical schema includes definitions for tables, columns, indexes, and many other attributes. The DDFs for a database contain the database's schema.

Table 3  Example Table for Food Preferences

<table>
<thead>
<tr>
<th>Col Names</th>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name</td>
<td>Meat</td>
<td>Grain</td>
<td>Drink</td>
</tr>
<tr>
<td>Row 1</td>
<td>Fred Black</td>
<td>sushi</td>
<td>wheat</td>
<td>sake</td>
</tr>
<tr>
<td>Row 2</td>
<td>Jane Doe</td>
<td>steak</td>
<td>oats</td>
<td>beer</td>
</tr>
<tr>
<td>Row 3</td>
<td>Ann Dean</td>
<td>cod</td>
<td>bran</td>
<td>spring water</td>
</tr>
</tbody>
</table>

**Remote**

The term remote refers to an object, such as a file server or a database, that is not located in the computer you are using now. When you connect to a database over the network, you are connecting to a remote database. Remote is the opposite of local. Remote can refer to either the client or the server, depending on whether you are currently seated at the server computer or a client computer. Remote always refers to an object that is not located on the system you are using.

**Local**

The term local refers to the computer you are using right now, or something stored on this computer. A local database is a database in which the data files are stored on the hard disk of the computer you are currently using. Local is the opposite of remote. Local can refer to either the client or the server, depending on whether you are currently seated at the server computer or a client computer.
Introducing Pervasive PSQL

**Relational**

The term relational refers to the storage of data in the form of related tables. The related tables allow relationships to be created between sub-sets of data.

For example, you can see that both our example tables contain the Name column, and some of the names are the same. Because we can cross-reference the names in the Phone table with the names in the Food table, we have the power to ask and answer such questions as, “What is the phone number of someone who likes steak?” We may also answer such questions as, “Which consumer profile purchased the most product B after buying product A?”

You can see how powerful relational data access is. The SRDE component of Pervasive PSQL provides full relational access to your data.

**Join**

A join refers to an association between columns of related tables. Typically, a join operation is part of a SELECT query, which is used to obtain information from related tables.

One unique feature of Pervasive PSQL is that it allows applications to access data through either the industry-standard relational method outlined above, or through an ultra-high-speed transactional or hierarchical method known as the Btrieve interface. In fact, Pervasive PSQL allows applications to use both access methods at the same time to access the same data.

**Transactional Interface**

The transactional interface is a high-performance, low-overhead access method, capable of handling updates, inserts, and deletes much faster than other database products.

Applications that use the transactional interface bypass the relational interface and communicate directly with the MKDE to maximize performance.

In the interest of performance, the transactional interface offers only basic security, including file passwords and encryption. It does not allow SRDE data access to bypass transactional security.
Relational Interface

The relational interface uses industry-standard ODBC to provide a rich environment for data definition, security, reporting, stored procedures, and universal application access without requiring any application programming. Databases that are ODBC-enabled can be accessed by any ODBC-standard software program.

As an end user of an application based on Pervasive PSQL, you may not be able to choose which access method your application uses, but your application vendor has most certainly taken this into account. No other DBMS available today offers this combination of flexible relational access and high-speed transaction throughput.

Additional Access Methods

In addition to the Transactional and Relational interfaces, Pervasive PSQL provides methods to access data through OLE DB, Java (JDBC), ActiveX, ADO, and Pervasive Direct Access Components (PDAC) for Delphi and C++ Builder.

Terminology Revisited

When using the Btrieve interface, the terms table and database are generally not used, and data files are referred to directly as such. In addition, Btrieve users normally use the terms records and fields rather than rows and columns.
Understanding the DBMS Products

Pervasive PSQL is available in different versions. The major differences between the versions are price, multi-user features, and the licensing model used.

- Pervasive PSQL Workgroup is the least expensive, but it provides support only for small workgroup environments.
- Pervasive PSQL Server is designed for maximum scalability in high-volume, mission critical database applications where there is a dedicated database server. The Server engine quickly becomes most economical as you increase the number of users.
- Pervasive PSQL Vx Server engine is designed for use in highly virtualized environments.

The database engine in all of the versions is fully compatible with any Pervasive PSQL database, and was designed with a common architecture. To upgrade from Workgroup to Server typically requires no changes to your application or to your database. Simply install the new version and you are ready to go. Because Pervasive PSQL Vx Server uses a different license model, it requires a full installation. You cannot upgrade Pervasive PSQL Server or Workgroup to Pervasive PSQL Vx Server.

**Pervasive PSQL Workgroup**

Pervasive PSQL Workgroup offers a peer-to-peer network setup designed for stand-alone single-user installations up to small workgroups. The Workgroup engine is the only engine that offers multi-user access to Pervasive PSQL data located on a computer where no database engine is installed.

A major difference between Workgroup and Server is the Gateway feature of Workgroup. When there is no database engine running on the computer where the data is located, normally the first database engine to connect to that data handles all requests from other engines to access that data. This feature can be configured so that the same Workgroup engine always services that data, or the Gateway designation can be allowed to “float” based on which Workgroup engine connects to the data first during any given work day.

The Workgroup product uses the user count licensing model. See License Models.
**Pervasive PSQL Server**

Pervasive PSQL Server offers a full client/server architecture providing excellent performance and scalability for up to thousands of concurrent users. The Server engine can be monitored and configured remotely.

The Server engine must be located on the same computer as the data files it is intended to access.

Pervasive PSQL Server uses the user count licensing model. See License Models.

**Pervasive PSQL Vx Server**

Pervasive PSQL Vx Server is designed to support highly virtualized environments. For instance, Vx Server includes support for Cloud computing (private, community, and hybrid), as well as for full virtualization, partial virtualization, and paravirtualization.

Pervasive PSQL Vx Server has no restrictions on hosting, connection pooling, Internet or intranet use. No additional license is required for use with hypervisor features such as live migration, failover, fault tolerance (FT), high availability (HA), and disaster recovery.

Pervasive PSQL Vx Server remains authorized and fully functional provided that the MAC addresses and hostname remain consistent and Internet connectivity is maintained.

Pervasive PSQL Vx Server uses the capacity-based licensing model. See License Models.

Because Pervasive PSQL Vx Server is built on technology used for Pervasive PSQL Server, it shares much of the database functionality and documentation provided for Pervasive PSQL Server.

Throughout the Pervasive PSQL documentation, all topics pertaining to Pervasive PSQL Server also apply to Pervasive PSQL Vx Server unless differences are explicitly called out. For convenience of reference, the Pervasive PSQL Vx Product Guide contains content specific to Pervasive PSQL Vx Server.

**Crystal Reports for Pervasive PSQL**

Crystal Reports is an optional product that provides rich capabilities for creating and formatting reports based on Pervasive PSQL databases. Reports can be customized in thousands of ways and published as HTML, Microsoft Word document, Microsoft Excel document, or other formats.

For more information about Crystal Reports, contact your sales representative or visit the Pervasive web site at http://www.pervasivedb.com.
Introducing Pervasive PSQL

Using Pervasive PSQL Documentation

All Pervasive PSQL documentation assumes you are familiar with the basics of using a computer, such as clicking and dragging, opening and saving files. If you need assistance with these tasks, please consult the documentation that came with your computer and/or operating system.

The viewer for the user documentation is integrated into Pervasive PSQL Control Center (PCC). Access the documentation through the PCC interface on the Welcome view, in the Help menu, by pressing F1 (Windows) or Shift F1 (Linux).

The following is a summary of the most commonly used books in the documentation library. The library contains other books, all of which can be accessed through the PCC interface as explained above.

Getting Started

Getting Started with Pervasive PSQL helps you get Pervasive PSQL running with installation, setup, and troubleshooting information. Getting Started With Pervasive PSQL covers the following topics:

- Preparing to install Pervasive PSQL v11 SP3
- Installing Pervasive PSQL v11 SP3
- Upgrading from previous versions of Pervasive PSQL or Btrieve
- Configuring Pervasive PSQL v11 SP3
- Troubleshooting your Pervasive PSQL v11 SP3 installation
- Where to go for Pervasive PSQL product information and technical support

What Is New

What Is New in Pervasive PSQL provides an overview of the new features and changed behavior for the current release of Pervasive PSQL relative to the most recent previous release. This book provides summary information on new features and directs you to the locations in the documentation where the new features are fully discussed.
Using Pervasive PSQL Documentation

**User's Guide**

Pervasive PSQL User's Guide introduces Pervasive PSQL and describes common user tasks. The guide discusses the database engine, Pervasive PSQL utilities and other key components; the differences between Server and Workgroup engines; and the differences between ODBC and Btrieve access. Pervasive PSQL User's Guide provides you with the basics to work with Pervasive PSQL successfully.

**Advanced Operations Guide**

Advanced Operations Guide provides detailed information at the administrative level, including the steps to perform common procedures and several new ones. Topics include:

- checking database consistency
- performing periodic backups
- configuring network protocols and understanding network topologies
- working with database security
- basic configuration guidelines
- configuration options reference
- moving, renaming, compacting and rebuilding files

**SQL Engine Reference**

SQL Engine Reference gives database programmers a complete reference guide to the SQL relational database language. It also covers SQL engine parameters and limitations.

**Status Codes and Messages**

Status Codes and Messages documents all possible status codes and numbered messages that can be received when using Pervasive software.

The Status Codes Quick Reference is also included with your complete documentation set.

**Pervasive PSQL Vx Product Guide**

For convenience of reference, documentation for Pervasive PSQL Vx Server is consolidated in the Pervasive PSQL Vx Product Guide. However, because Pervasive PSQL Vx Server is built on technology used for Pervasive PSQL Server, it shares much of the database functionality and documentation provided for Pervasive PSQL Server. Throughout the Pervasive PSQL documentation, all topics pertaining to Pervasive PSQL Server also apply to Pervasive PSQL Vx Server unless differences are explicitly called out.
Introducing Pervasive PSQL

**Additional Information**

You can download the product documentation, white papers, technical papers, and view the online library at http://www.pervasivedb.com.
File System Security

The Pervasive PSQL engine adheres to the file system security defined by the specific operating system (OS), such as Windows File Sharing.

Only the Server engine can enforce OS-level file security based on the privileges assigned to the login user name. The Workgroup product does not attempt to do this. In a small office, where Workgroup engines are most common, this can be considered a plus because they are usually short on networking experts, and the fewer barriers to successful data access the better.
Introducing Pervasive PSQL
This chapter covers the basic tasks you need to know to work with Pervasive PSQL databases. Included in this chapter are the following sections:

- Starting and Stopping the Database Engine
- Granting Administrative Rights for the Database Engine
- Setting Up ODBC Database Access
- Setting up Database Access with PCC
- Accessing Data on a Remote Engine Using PCC
- Accessing Data via ODBC From Other Applications
- Deleting DSNs
- Using the Fast User Switching Feature of Windows XP
Starting and Stopping the Database Engine

This section outlines how to start and stop the Pervasive PSQL engine. Some engine configuration parameters, you need to stop and restart the engine in order for a particular change in your configuration to take effect.

To start and stop the database engine, follow the instructions for your engine and platform:

- Starting and Stopping the Server Engine on a Windows Server
- Starting and Stopping the Workgroup Engine on Windows
- Starting and Stopping the Database Engine on Linux

Starting and Stopping the Server Engine on a Windows Server

On Windows server environments, Pervasive PSQL Server runs as services. The services are loaded as part of the installation process and are set to be always available if you followed the default installation.

Services Dependencies

Additional Pervasive PSQL products such as Pervasive DataExchange also install services. Pervasive DataExchange depends on both Pervasive PSQL Transactional Engine and Pervasive PSQL Relational Engine services.

The following table summarizes the behavior of the dependent services for start, stop, and restart actions of the database engine services.
Starting and Stopping the Database Engine

Note that the behavior of the dependent services is the same regardless with which application you start, stop, or restart the database engine services (PCC, Windows Services, Net Start, PSC).

Table 4  Action of Dependent Services Based on Action of Database Engine Services

<table>
<thead>
<tr>
<th>Database Engine Service</th>
<th>Start</th>
<th>Stop</th>
<th>Restart</th>
<th>Service Action for Data Exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pervasive PSQL Transactional Engine</td>
<td>✓</td>
<td></td>
<td></td>
<td>no action</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>Restart</td>
</tr>
<tr>
<td>Pervasive PSQL Relational Engine</td>
<td>✓</td>
<td></td>
<td></td>
<td>no action</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>Restart</td>
</tr>
</tbody>
</table>

If you start the service for DataExchange, the Pervasive PSQL Relational Engine starts as a prerequisite service.

Note: The dependent services stop without displaying a message that they will be stopped.

➤ To start the database services on a Windows server using PCC

See To start or stop services.

➤ To stop the database services on a Windows server using PCC

See To start or stop services.
To start the database services on a Windows server using Control Panel

1. In the Windows Control Panel, double-click Administrative Tools then double-click Services.

   A dialog box similar to Figure 1 appears.

![Figure 1 Services Dialog Box](image)

2. Right-click Pervasive PSQL Transactional Engine then click Start. Right-click Pervasive PSQL Relational Engine then click Start.

To stop the database services on a Windows server using Control Panel

1. In the Windows Control Panel, double-click Administrative Tools then double-click Services.

   A dialog box similar to Figure 1 appears.
Starting and Stopping the Database Engine

2 Right-click **Pervasive PSQL Relational Engine** then click **Stop**.
Right-click **Pervasive PSQL Transactional Engine** then click **Stop**.

➢ To start the database services on a Windows server by using Net Start or PSC

1 Run one of the following commands at the operating system:

   a. `net start "Pervasive.SQL (<transactional | relational>)"`
   b. `psc start "Pervasive.SQL (<transactional | relational>)"`

➢ To stop the database services on a Windows server by using Net Stop or PSC

1 Run one of the following commands at the operating system:

   a. `net stop "Pervasive.SQL (<transactional | relational>)"`
   b. `psc stop "Pervasive.SQL (<transactional | relational>)"

Starting and Stopping the Workgroup Engine on Windows

➢ To start the Workgroup Engine as an application on Windows

These steps assume that the Workgroup Engine was installed as an application. See Installing the Pervasive PSQL Workgroup for Windows in Getting Started With Pervasive PSQL.

1 Click **Start Workgroup Engine** from the operating system **Start** menu or **Apps** screen.
To stop the Workgroup Engine as an application on Windows

These steps assume that the Workgroup Engine was installed as an application. See Installing the Pervasive PSQL Workgroup for Windows in Getting Started With Pervasive PSQL.

1. Click **Stop Workgroup Engine** from the operating system **Start** menu or **Apps** screen.

   **Note** You will receive a warning message when trying to stop the engine if any of the following is true:

   - There are active clients.
   - No activity took place since the engine loaded.
   - 10 seconds has not elapsed since the last operation took place.

To start the Workgroup Engine as a service on Windows

These steps assume that the Workgroup Engine was installed as a service. See Installing the Pervasive PSQL Workgroup for Windows in Getting Started With Pervasive PSQL.

1. Refer to the various methods described in the section Starting and Stopping the Server Engine on a Windows Server.

2. Substitute “Pervasive PSQL Workgroup Engine” as the service name if you use Windows Services control panel. Substitute “psqlWGE” as the service name if you use net start or psc.

To stop the Workgroup Engine as a service on Windows

These steps assume that the Workgroup Engine was installed as a service. See Installing the Pervasive PSQL Workgroup for Windows in Getting Started With Pervasive PSQL.
Starting and Stopping the Database Engine on Linux

1 Refer to the various methods described in the section Starting and Stopping the Server Engine on a Windows Server.

2 Substitute “Pervasive PSQL Workgroup Engine” as the service name if you use Windows Services control panel. Substitute “psqlWGE” as the service name if you use net start or psc.

In Linux, the database engine runs as a daemon. The daemon is loaded as part of the installation process and is set to be always available if you followed the complete installation.

You must be logged in as the root user to start and stop the Pervasive PSQL v11 SP3 daemon process. Use the shell script psql to start and stop the process.

➢ To start the database engine on Linux

Enter the following at the command line:

/etc/init.d/psql start

For Pervasive PSQL 64-bit Server, this command starts both the transactional daemon and the relational daemon. The relational daemon is 32-bit only.

➢ To stop the database engine on Linux

Enter the following at the command line:

/etc/init.d/psql stop

For Pervasive PSQL 64-bit Server, this command stops both the transactional daemon and the relational daemon.

➢ To force stop the database engine on Linux

If the database engine does not start or stop correctly using the psql start and stop commands, you may need to force stop the engine. The force stop removes any shared memory and semaphores created by Pervasive PSQL and stops the Pervasive PSQL relational and transactional processes. Enter the following at the command line to force stop the database engine:

/etc/init.d/psql force
Granting Administrative Rights for the Database Engine

This section begins by outlining those Pervasive PSQL v11 SP3 tasks that require administrative-level access at the operating system level and those that do not. The section then walks you through the steps to grant a user administrative-level access for each of the supported operating systems.

**Note** This section only applies to the Server engine unless otherwise noted.

<table>
<thead>
<tr>
<th>Tasks Requiring Administrative Rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative-level rights are required to:</td>
</tr>
<tr>
<td>- create and configure named databases and tables</td>
</tr>
<tr>
<td>- view or modify a table design with Table Editor</td>
</tr>
<tr>
<td>- set engine configuration options</td>
</tr>
<tr>
<td>- view and set engine monitoring values</td>
</tr>
<tr>
<td>- view certain engine configuration settings</td>
</tr>
<tr>
<td>- restarting the engine when running as a service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How Administrative Rights are Granted</th>
</tr>
</thead>
<tbody>
<tr>
<td>To have administrator-level access you must either:</td>
</tr>
<tr>
<td>- possess full administrator-level rights on the machine on which the database engine is running (a domain administrator, for example, may lack full permissions on certain local machines)</td>
</tr>
<tr>
<td>or</td>
</tr>
<tr>
<td>- be a member of the operating system group Pervasive_Admin.</td>
</tr>
<tr>
<td>To modify a table design with Table Editor, you must have full administrator rights on the machine on which the database engine is running even if you are a member of the Pervasive_Admin group.</td>
</tr>
</tbody>
</table>

**Note** For Linux servers, administrator-level rights can be granted only by using the btadmin utility to add users and passwords to the btpasswd file.
Granting Administrative Rights for the Database Engine

The Pervasive_Admin option is offered so that you can grant users administrative rights to the database engine without granting them administrative rights to the operating system where the database engine resides.

Rights Within an Active Directory Environment

You may use one or more Pervasive_Admin groups within an Active Directory environment. See Active Directory Service in Getting Started With Pervasive PSQL.

Rights Provided to non-Administrative Users

Runtime-only access enables a user without administrator-level rights to perform such functions as:

- extract a list of DSNs
- extract a count of DSNs
- extract information on a DSN
- extract information on the location of the DBnames configuration file (dbnames.cfg)
- connect to databases
- retrieve, update, insert, and delete data (as permitted by database security)

Tasks for Granting Administrative Rights

To grant a user administrative rights, follow the instructions for your platform:

- Granting Administrative Rights on a Windows Server
- Granting Administrator Rights on Linux
- Logging in as Administrator on any platform

Granting Administrative Rights on a Windows Server

Users who are members of Pervasive_Admin or of Administrators are permitted to perform administrative tasks on the database engine.

To grant a user database administrator rights on a Windows 32-bit Server Platform

Note You must be logged onto the Windows server as a user with full administrator-level rights on the server or be a member of the Pervasive_Admin group defined on the server.
Using Pervasive PSQL

1. In the Windows Control Panel, double-click Users and Passwords.

2. Click the Advanced tab. In the Advanced User Management area, click Advanced.

3. Click the Groups folder. From the menu, click Action → New Group.

4. Type in Pervasive_Admin as the group name.
   (To add users to this group, click Add, select user name, click Add then OK.)

5. Click Create to create the group.

6. Click Close.

Note: If the Log on as setting for the Pervasive PSQL services is not System Account, see Services Settings and Log In Authority.

Services Settings and Log In Authority

Certain operating system settings for the Pervasive PSQL services must be in effect for you to log in to the machine running the database engine. These settings apply whether or not you use a Pervasive_Admin user group.

The settings apply to the Pervasive PSQL Server engine and to the Workgroup engine if you are running the Workgroup engine as a service. See Running the Workgroup Engine as a Service in Getting Started With Pervasive PSQL.

Default Setting

By default installation, both the Transactional and Relational services set Log on to Local System Account.
Granting Administrative Rights for the Database Engine

Logging On as “This Account”

If you change the Log on as setting to This account, you must change the user rights policy Act as part of the operating system for the account. Otherwise, remote login fails.

For example, the Monitoring utility requires that you log in to the operating system on the machine where the database engine is running. You will receive a message that login failed if the account specified for This account cannot act as part of the operating system.

Note that even the Administrator account requires that you set the user rights policy for Act as part of the operating system.

You specify This account on the services property sheet.

User Rights Policy Tasks

The following tasks explain how to change the user rights policy.

➢ To Set User Rights Policy on Windows 32-bit Platforms

1. Access the operating system Control Panel.
2. Double-click Administrative Tools.
3. Right-click Local Security Policy, then click Open.
Using Pervasive PSQL

4. Expand the tree for Local Policies, and click on User Rights Assignment.

5. In the policy pane, right-click Act as part of the operating system, then click Security.

6. Click Add.

7. In the Name pane, click on the user or group for whose account you want to grant the user policy. (For example, you could grant the policy to the Pervasive_Admin group.)

8. Click Add.

The user name is added to the bottom pane. For example, the following image shows that the Administrator has been added.
Granting Administrative Rights for the Database Engine

9 Click OK.

The user name is added to the settings for local security policy.

10 Click OK.

11 Exit the window for Local Security Settings, then exit Administrative Tools.

Granting Administrator Rights on Linux

To grant a user administrator rights on Linux

A user cannot remotely administer a server engine on Linux unless the user has first been set up as a database user with administrative rights. You can perform this task by using the btadmin utility at the server command line.

See also Pervasive PSQL Account Management on Linux in Getting Started With Pervasive PSQL for a complete discussion of configuring the environment for administrative rights.

1 Login to the Linux server as psql (or as root if the PATH and LD_LIBRARY_PATH variables have been set and exported). No other user is permitted to run btadmin.

2 Create a new user with administrative rights by running btadmin:

btadmin -p passwd a+ user_name

For example, if you wanted to create an administrative user “tim” with password “tim56,” you would enter the following command:
Using Pervasive PSQL

btadmin -p tim56 a+ tim

**Note** Users created with btadmin are not related to Linux system users. These users are known only to the database engine.

Logging in as Administrator on any platform

➢ **To connect to a remote Pervasive PSQL v11 SP3 server**

1. Use the Monitor utility to connect to a remote server. Refer to the Advanced Operations Guide for a discussion of this utility.

2. Enter your operating system user name and password, and click **OK**.

![Figure 3: Connect to Remote Server Dialog Box](image)

The password is encrypted before being sent over the network using a unique and pre-defined encryption key. The Pervasive PSQL v11 SP3 engine unpacks and decrypts the user name and password, and verifies access. It then returns a status code to the client indicating the success or failure of the verification.
Setting Up ODBC Database Access

This section reviews some conceptual information on setting up ODBC access to your database.

Topics covered include the following basic concepts:

- **ODBC Standard**
- **Servers and Clients**
- **Data Source Names**
- **Internal Database Name**
- **Applications Using the Transactional Interface**

**Note** The Pervasive PSQL Java utilities do not require DSNs. The Pervasive PSQL Control Center, for example, uses JDBC not ODBC. DSNs are required only if your application uses ODBC to access the database.

**ODBC Standard**

Pervasive PSQL adheres to the Microsoft standard for ODBC database connections. According to the standard, applications using ODBC must connect to databases through Data Source Names (DSNs) defined in the operating system.

**Note** Pervasive PSQL does not support File DSNs. You must use User or System DSNs. System DSNs are generally preferred, because they are available to all users on a given computer.

Every Pervasive PSQL database that you expect to access using an ODBC application must have a DSN that identifies the database. A DSN that points to the Pervasive PSQL database engine is called an Engine DSN. Engine DSNs are 32-bit only and are deprecated. A 32-bit DSN that points to an Engine DSN or to a named database is called a Client DSN. A DSN on a 64-bit operating system is simply referred to as a 64-bit DSN, without designation of Engine or Client.

New or revised 32-bit applications, local or remote, should connect to a named database instead of using Engine DSNs. Alternately, applications could use DSN-less connections by specifying the Pervasive PSQL ODBC driver name (see Pervasive PSQL ODBC...
Driver Names in SQL Engine Reference). Avoiding the use of Engine DSNs positions your application for the future when Engine DSNs will no longer be supported in Pervasive PSQL.

Windows 64-bit operating systems contain two different executable files for ODBC Administrator, one for 32-bit DSNs and one for 64-bit DSNs. Each ODBC Administrator lists the system DSNs that only match its bitness. For the 64-bit Pervasive PSQL Server or Workgroup, PCC contains separate options in the Tools menu to start the 32-bit or the 64-bit ODBC Administrator. If an ODBC Administrator is already open, Windows defaults to it. That is, if the 32-bit ODBC Administrator is open and you attempt to start the 64-bit one, Windows displays the 32-bit version (and vice versa). This is a limitation of the Windows operating system, not Pervasive PSQL.

Figures 4 shows possible DSN configurations.

Figure 4  Example DSN Configurations

Servers and Clients

Pervasive PSQL servers are also clients. The client components of Pervasive PSQL are installed with every Server engine or Workgroup engine. So you can use your server machine to connect to other servers as a client. Pervasive PSQL clients can connect to remote machines where a Pervasive PSQL Server engine is installed.
Setting Up ODBC Database Access

**Data Source Names**

The ODBC client-server architecture calls for the naming of each specific data set so that it can be referred to by a well-known name. There are generally three ways to create DSNs:

1. Create a DSN from the server console.
2. Create a DSN remotely from a client machine.
3. Create a Client DSN on each client machine.

While Pervasive tools can access remote databases without a client DSN present on the client machine, ODBC-based applications such as Microsoft Excel and Microsoft Access cannot do so. You must create a client DSN on each client computer that needs to access network databases from local ODBC applications.

**Internal Database Name**

The method used by Pervasive PSQL to identify a database is an internal Database Name (DBNAME). If you are using ODBC to access the database, you need to create a Data Source Name (DSN) entry that refers to one DBNAME. You may set up more than one DSN that refers to the same DBNAME. If the physical location of the data files on the server is changed, only the DBNAME needs to be updated. All DSNs remain unchanged.

**Applications Using the Transactional Interface**

Pervasive PSQL databases that are accessed only through the transactional interface do not need DSNs. However, the database tables are not visible in PCC nor can they be manipulated using PCC. To view table data with PCC in readable form, the table must be defined in the data dictionary files (DDFs). See DDF Builder User's Guide.

Also, databases created with the Pervasive PSQL Java utilities do not require DSNs. The Pervasive PSQL Control Center, for example, uses JDBC not ODBC.
Setting up Database Access with PCC

You must know the name of the server where the database is located. If the database already has a DBNAME, PCC uses it. If you wish to create a new database but use existing data files, you must know the location of the data files on the server.

To create a DSN on a remote machine, you must possess administrator rights on the remote machine that houses the database you wish to access. You must have OS system rights to create a System DSN on the local machine.

Setting Up Database Access on Windows

➢ To set up database access on Windows
1. Follow the steps listed in To register a remote server engine. Existing databases with a DBNAME on the registered server can then be accessed from PCC.
2. Optionally, follow the steps listed in To create a new database. The new database can then be accessed from PCC.

➢ To set up ODBC database access on Windows
1. Follow the steps listed in To register a remote server engine. Existing databases with a DBNAME and a DSN on the registered server can then be accessed from PCC.
2. Optionally, follow the steps listed in To create a new database and ensure that the Create 32-bit Engine DSN option is selected.

By default, PCC creates a 32-bit system DSN with the same name as the database name. The new database has a DSN associated with it and can be accessed through ODBC.

➢ To set up Engine DSNs using ODBC Administrator

Note that the ODBC Interface GUIs for 32-bit and 64-bit Windows operating systems are different. The one for 32-bit DSNs allows you to specify an Engine DSN. The 64-bit interface driver supports only named databases. The 64-bit Client Interface can connect to a local named database, thus replacing the function of the Engine DSN, or to a remote named database. Connection to an Engine DSN is not supported.
Setting up Database Access with PCC

Because 32-bit Engine DSNs have been deprecated, Pervasive recommends that new or revised 32-bit applications connect to a named database through a Client DSN or use a DSN-less connection by specifying “Pervasive ODBC Client Interface.” See To set up a Client DSN using ODBC Administrator.

1. In PCC, click Tools then ODBC Administrator.
2. Click the System DSN tab, then Add.
3. In the list, click Pervasive ODBC Engine Interface.
4. Click Finish.
   The Pervasive ODBC Engine Interface dialog displays.
5. Type a Data Source Name.
6. For Database Name, select in the list for which you want to create the Engine DSN.
7. Click Advanced if you want to specify advanced connection attributes.
   See Advanced Connection Attributes for Engine DSN in SQL Engine Reference.
8. Click OK.
9. Click OK.

To set up a named database on a Linux server

Database names are created in Linux by using the dbmaint utility at the server. For a complete description of dbmaint, see dbmaint or read the dbmaint man page.

Note This utility can only be run by user accounts belonging to group pvsw. See Pervasive PSQL Account Management on Linux in Getting Started With Pervasive PSQL.

1. To create an empty database, use the following at the command line:
Using Pervasive PSQL

```
dbmaint a | d | l [-b] [-i] [-e] -n Dbname [-ldictpath] [-ddatapath]
```

The list of commands for dbmaint include:

- **a** - add database name
- **d** - delete database name
- **l** - list all database names

Options include:

- **-b** - create Bound database
- **-i** - create database with Relational Integrity enforced
- **-e** - do not create dictionary files for database
- **-n Dbname** - specify database name
- **-ldictpath** - specify dictionary path
- **-ddatapath** - specify data path
- **-a** - show full data in the DBNames list

For example, to create DBName TEST with relational integrity, type:

```
dbmaint a -i -n TEST
```

**Note** Unless datapath is specified, the new database is created in the default location, $PVSW_ROOT/data. Likewise, if dictpath is not specified, the dictionary is created in the default location.

- To delete an existing database, use the following at the command line:
  
  ```
dbmaint d -n Dbname
  ```
  
  For example, to delete the newly created database TEST, type:
  
  ```
dbmaint d -n TEST
  ```

- To list all existing databases:
  
  ```
dbmaint l [-a]
  ```

2 Set up DSNs using the dsnadd utility. See dsnadd.

---

**Setting Up Client Access from a Windows Client**

1 **To enable client access to a remote Pervasive PSQ database**
   
   Access **Control Center** from the operating system **Start** menu or **Apps** screen.
2 In the Pervasive PSQQL Explorer pane, right-click Engines then click New then Server.

The Pervasive PSQQL Explorer pane is the column on the left side of the window that contains a list of machines to which you are connected.

Note The machines listed in your Pervasive PSQQL Explorer will remain between sessions. To remove a machine, right-click the machine name and click Delete.

3 Enter the Server name where the Pervasive PSQQL v11 SP3 database engine resides.

You need to be authenticated on the remote engine, and a dialog displays prompting you for a user name and password.

4 Enter the user name and password in the appropriate fields and click OK.

You are now connected to the remote Pervasive PSQQL engine.

➢ To set up a Client DSN using ODBC Administrator

1 In PCC, click Tools and select ODBC Administrator.

Note Windows 64-bit operating systems contain two different executable files for ODBC Administrator, one for 32-bit DSNs and one for 64-bit DSNs. Each ODBC Administrator lists the system DSNs that only match its bitness. For the 64-bit Pervasive PSQQL Server or Workgroup, PCC contains separate options in the Tools menu to start the 32-bit or the 64-bit ODBC Administrator. If an ODBC Administrator is already open, Windows defaults to it. That is, if the 32-bit ODBC Administrator is open and you attempt to start the 64-bit one, Windows displays the 32-bit version (and vice versa). This is a limitation of the Windows operating system, not Pervasive PSQQL.

2 Click the System DSN tab, then Add.
Using Pervasive PSQL

**Note** Pervasive PSQL does not support File DSNs. You must use User or System DSNs. System DSNs are generally preferred, because they are available to all users on a given computer.

3 In the Drivers window, select **Pervasive ODBC Client Interface** for a 32-bit DSN or **Pervasive ODBC Interface** for a 64-bit DSN.

4 Specify a **Data Source Name** (DSN) for which you want to set up a connection. This DSN helps you identify the data source.

![Pervasive ODBC Interface for Client DSNs](image)

Note that on the 64-bit interface, the Engines DSN option is **not** available and is not present on the GUI.

5 For **Server Name/IP**, specify a machine name or TCP/IP address of a machine running the Pervasive PSQL database engine.

6 Select the Transport Hint desired.

7 Specify a **Database Name** to which to connect. Click **Get List** to obtain a list of the existing databases on the server listed for **Server Name/IP**.

For 32-bit DSNs only, you may specify an Engine DSN instead of a database name. (Note that Engine DSNs are deprecated, so it is preferable to specify a database name rather than an Engine DSN.) Click **Get List** to obtain a list of the existing DSNs on the server listed for **Server Name/IP**. Click **Modify** to modify an existing DSN, or **Create** to create a new DSN.
8. If you want advanced connection attributes, click **Advanced** and make your selections in the dialog box that appears.

![Advanced Connection Attributes for Client DSNs](image)

See [Advanced Connection Attributes for Client DSN](#) in SQL Engine Reference for details about the attributes. Do not modify the **Network** or **TCP/IP Port Number** attributes unless you first review the information for them.

Click **OK** to accept the attribute selections and to return to the ODBC interface dialog.

9. Click **OK**.

10. You can now set up another DSN or exit the ODBC Administrator.

**Setting Up a Client DSN on a Linux Client**

Use the dsnadd utility. See dsnadd.
Accessing Data on a Remote Engine Using PCC

You can use Pervasive PSQL Control Center to access data on a remote machine on which a Pervasive PSQL v11 SP3 engine is installed.

Tip You will need to login as an administrative user on the remote engine to perform most functions. This means that you must have full administrator-level rights on the remote server or be a member of the Pervasive_Admin group defined on the remote machine.

To access data on a remote Pervasive PSQL engine

1 In the Pervasive PSQL Control Center, double-click the remote Pervasive PSQL Engines node, then double-click Databases.

2 In the Databases list double-click DEMODATA, then double-click Tables.
3 Double-click the **Dept** table from the **Tables** list.

![Figure 8: Selecting the Dept Table in DEMODATA](image)

By default, a “SELECT * FROM” query is run and the table results are displayed in an active grid as shown in Figure 9.

The data displayed in the active grid that loads is updateable. That is, changes you make to the data in that grid are stored to the database.

![Figure 9: Displaying the Dept Table in DEMODATA](image)

4 Refine the query to restrict the results to only departments that start with the letter ‘M’ by altering the query at the top half of the screen with the following statement:

```
SELECT * FROM Dept WHERE Name LIKE 'M%'
```
Using Pervasive PSQL

5 Click the **Execute in Grid** toolbar button or press F9 to display the results of the revised query shown in Figure 10.

Figure 10  Refining Your Query - Dept Table in DEMODATA

You have now obtained data from the remote database engine.

For information on advanced operating and maintenance tasks, including database operations, see the Advanced Operations Guide.
Accessing Data via ODBC From Other Applications

This section explains how to access data using Microsoft Access and Microsoft Excel.

The examples covered in this section are:

- Accessing Data Using Microsoft Excel
- Accessing Data Using Microsoft Access

Before You Begin

Does the Database Have a DSN Available?

- If you are connecting from a client workstation or from a Workgroup workstation to a server, you must have a Client DSN defined on your workstation for the given remote database. Information on how to create a Client DSN is provided in Setting Up Client Access from a Windows Client.
- If you have a Workgroup engine installed on your computer, you may have a DSN defined on your computer for either local or remote databases. Information on how to create a DSN is provided in Setting Up Database Access on Windows.

Note The instructions in this section apply only to Pervasive PSQL v11 SP3, not to previous versions.

Accessing Data Using Microsoft Excel

To access Pervasive data using Excel

Tip You must have the Pervasive PSQL client or any version of the Pervasive PSQL engine installed on the computer where you are using Excel.

1 Start Excel.
2 From the Data menu, choose: Get External Data → New Database Query as shown below.
3 The **Choose Data Source** box lists the defined data sources for any ODBC drivers that are installed on your computer. From this list, click on the Client or Server DSN for the Pervasive database you wish to access, as shown in the example below.

![Figure 11: Accessing Pervasive Data using Microsoft Excel](image)

If the database you want does not appear in the ODBC Source list, see *Before You Begin*.

4 Click **OK**. You may be prompted to login to the Pervasive PSQL database. If the database is not secure, leave the **User** and **Password** fields empty. Otherwise enter your assigned user name and password.

5 The **Query Wizard** opens. Simply follow the wizard to select your options such as which tables to query, how to filter and sort the data, and how you would like Excel to return the Pervasive data to you for your use.
Accessing Data via ODBC From Other Applications

**Accessing Data Using Microsoft Access**

1. Open Microsoft Access.
2. From the Access dialog box, choose **Blank Access database** as shown below. Click **OK**. (Note that you may also add Pervasive PSQL tables to an existing Access database.)

![Create a New Database using Microsoft Access](image)

3. Next, the **File New Database** dialog box opens and asks you to name the new database. Name the database and click **Create**.
4. From the Access menu, choose: **File** ➔ **Get External Data** ➔ **Link Tables**.

---

**Note** You have the option to **Import** data or **Link Tables** to the new database. When you choose **Import**, you break the link to the ODBC data source immediately following the import procedure. Essentially, **Import** creates a static copy of the data. When you choose **Link Tables**, Microsoft Access keeps the connection open and remains dependent upon the ODBC data source each time the data is accessed. This way, the data you see reflects any changes to the data at its source.
Note: If you wish to link to a file on a local area network, make sure to use a universal naming convention (UNC) path, instead of relying on the drive letter of a mapped network drive in Windows Explorer. A drive letter can vary on a computer or may not always be defined, whereas a UNC path is a reliable and consistent way for Microsoft Access to locate the data source that contains the linked table.

5 In the Link dialog box, in the Files Of Type box, select ODBC Databases.

6 The Select Data Source box lists the defined data sources for any ODBC drivers that are installed on your computer. Click the Machine Data Source tab as shown in the next figure.
Select the ODBC data source that you want to link. If the ODBC data source that you selected requires you to log on, enter your user name and password (additional information might also be required), and then click **OK**.

**Note** To define a new data source for any installed ODBC driver, click **New**, and then follow the instructions in the **Create New Data Source** dialog box and the dialog boxes that follow it before proceeding.

**Tip** If you are linking a table, select the **Save The Login ID And Password** check box to store the information for the table in the current database, so that users will not have to enter it each time. If you leave the check box cleared, all users must enter the logon ID and password every time they open the table with Microsoft Access in each new session. Your network administrator can also choose to disable this check box, requiring all users to enter a user name and password each time they connect to the database.

If the database you want does not appear in the ODBC Source list, see **Before You Begin**.
The Access **Link Tables** dialog box opens. Click each table that you want to import or link, and then click **OK**.

**Note** Microsoft Access cannot display more than 256 columns in a table. If you need to display more than 256 columns, you may wish to use a different tool.

Linking to your Pervasive data is complete. As shown in the figure below, Access presents you with options for designing the new database. View the linked tables by double-clicking on the table name.

**Figure 16** Using Pervasive Data in Microsoft Access

**Note** If you are linking a table and it does not have an index that uniquely identifies each record, then Microsoft Access displays a list of the fields in the linked table. Click a field or a combination of fields that will uniquely identify each record, and then click **OK**.
Deleting DSNs

The procedures in this section do not delete Data Dictionary Files (DDFs) or data files.

By default, when you delete a database in the Pervasive PSQL Control Center, the associated DSN entries are removed simultaneously.

➢ To toggle automatic removal of DSN entries when deleting a database in PCC

1. On the PCC Window menu, click Preferences. Expand the Pervasive node if it is not already expanded.

2. Click General.

3. Clear the option Always remove associated DSN entries and click OK.

When you delete a database in PCC, you will be prompted by the Confirm DSN Removal dialog before deleting it.

![Confirm DSN Removal dialog]

You can clear those DSNs that you do not want to delete.

4. You can turn on automatic removal of DSN entries at anytime by selecting Always remove associated DSN entries in:
   - Confirm DSN Removal dialog
   - General screen of the Preferences dialog.
To Delete a DSN using ODBC Administrator (Windows platforms only)

1. In PCC, click Tools and select ODBC Administrator.
2. In the ODBC Administrator window, click the System DSN tab.
3. Select the DSN you wish to remove, and click Remove.
   You are prompted to confirm removal of the DSN.
4. Click Yes.
5. After the DSN has been removed, click OK to exit ODBC Administrator.

If you are simply deleting an unwanted DSN, you are finished. If you need to re-create the DSN, you should refer to one or more of the following sections:

<table>
<thead>
<tr>
<th>If you need to do this</th>
<th>Refer to this section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re-create an Engine DSN on a Server engine or a Workgroup engine</td>
<td>One of:</td>
</tr>
<tr>
<td></td>
<td>• Setting Up Database Access on Windows</td>
</tr>
<tr>
<td></td>
<td>• Setting Up Database Access on a Linux Server</td>
</tr>
<tr>
<td>Re-create a Client DSN on a client workstation</td>
<td>Setting Up Client Access from a Windows Client</td>
</tr>
</tbody>
</table>
Using the Fast User Switching Feature of Windows XP

Fast user switching is a feature of Windows XP Home Edition and Windows XP Professional that allows you to switch between users without logging off from the computer. Multiple users can share a computer, switching back and forth among users without closing the programs each user is running. The users are all local to the computer, not logged in via a network. Only one user at a time can use the computer interactively.

(The Pervasive PSQL Server engine is not supported on Windows XP Professional or Home Edition. This support will be available when Windows XP Server is released.)

As of this release of Pervasive PSQL, the following Microsoft restrictions apply when fast user switching is turned on. These are restrictions of the operating system.

- The computer cannot be logged on to a network domain.
- The Serial Keys accessibility feature will not work.
- Offline files, such as Windows XP Offline Documentation, must be disabled.

Fast user switching allows only two types of users, classified as administrators or limited. Only administrators can turn on or turn off fast user switching.

Windows XP Professional allows two modes of operation, local and remote desktop. Remote desktop uses an XP client to access an XP machine from a remote computer. In many respects, the remote desktop feature is similar to terminal services on other Windows 32-bit platforms.

(You may also use a Pervasive PSQL client on an XP machine to communicate with a Pervasive PSQL Server engine across a network. The client functions the same as it does on any other Windows platform supported by Pervasive PSQL.)

Fast User Switching in Local Mode with Pervasive PSQL

The following conditions apply when you use a Pervasive PSQL client and Workgroup engine in local mode. Local mode refers to a local client communicating with a local engine.

53
Only one instance of the engine may be run at a time. You cannot run separate copies of the engine within separate user sessions. If you attempt to start the engine that has already been started by another user, the engine will not restart. No error message appears and no tray icon appears.

- If started by a user, the engine runs in the session of the first user to start the engine. The operating system rights of the first user to start the engine determine access rights to the database files. For example, a limited user can start the Pervasive PSQL engine but cannot create a new database.

- If the Pervasive PSQL Workgroup engine is started as a service, the operating system rights of the System Account determine access rights to the database files. See Running the Workgroup Engine as a Service in Getting Started With Pervasive PSQL.

Changes made to a database in one session are available to users of the database in other sessions. For example, User A adds a record to Database 1. User B fast user switches to her session. User B sees the record in Database 1 added by User A.

A database accessed by one user's session can lock the database from other users' sessions. For example, User A has Database 1 open in the PCC. User B fast user switches to her session and attempts to add security to Database 1. User B is prevented from adding security to the database.

Caution If you run the Workgroup engine as a console application, the first user to start the engine should not stop the engine if other users are accessing the engine. In addition, the first user must not log off because this causes the engine to terminate.

As an alternative to running the engine as a console application, you may run the engine as a service. See Running the Workgroup Engine as a Service in Getting Started With Pervasive PSQL.
Pervasive PSQL Control Center (PCC) is an easy-to-use, graphical tool designed to help you create and manipulate databases and control your database engine. It allows you to access nearly all the functions of the product from one place. This chapter leads you on a tour of PCC to help you learn the interface and common operations launched from PCC.

The topics in this chapter include:

- An Overview of Pervasive PSQL Control Center
- Services on Windows Servers
- Database Engines
- Capacity Usage Viewer
- Monitor
- Databases
- New Database GUI Reference
- Pervasive PSQL Database Tasks
- Tables
- Data
- Triggers, Stored Procedures, User-defined Functions, and Views
- Groups, Users, and Security
- Configuration
An Overview of Pervasive PSQL Control Center

Pervasive PSQL Control Center (PCC) is an integrated framework in which users can connect to Pervasive PSQL engines, set up and modify databases and tables, query and update data, tune engine performance, and access the Pervasive PSQL documentation library.

PCC uses a file explorer-like motif—a tree of objects—referred to as the Pervasive PSQL Explorer. This tree of objects can be opened or expanded to reveal more detail. Examples of objects include engines, databases, tables, and users. The following figures illustrate PCC with several window views displayed. The Pervasive PSQL Explorer is the tree view on the left.

Figure 17   Pervasive PSQL Control Center on Windows Platforms
Figure 18   Pervasive PSQL Control Center on Linux Platforms

Note that the PCC graphical user interface (GUI) may look cosmetically different depending on your distribution of Linux.

**Installing PCC**

On Windows platforms, PCC is installed by default when you install a database engine or a client. See *Pervasive PSQL Optional Features* in Getting Started With Pervasive PSQL.

On Linux, PCC is included in the full install. See *Full Installations* in Getting Started With Pervasive PSQL.

**Starting PCC on Windows**

Access Control Center from the operating system Start menu or Apps screen. You may also run the executable file `pcc.exe`. 
Using Pervasive PSQL Control Center

**Starting PCC On Linux**

You start PCC by running the executable script file `pcc` from a command prompt. The script file is located, by default installation, in the `usr/local/psql/bin` directory.

We recommend that you start PCC from a command prompt and not by double-clicking on the script file using a file browser application. See Table 6, Troubleshooting Guide for Running PCC.

The following requirements must be met to start PCC on Linux.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pervasive PSQL server or client</td>
<td>A compatible Pervasive PSQL server or client must already be installed on the same machine.</td>
</tr>
<tr>
<td>X-Server access</td>
<td>The <code>xhost</code> command controls which clients can access X-Windows on the current machine. By default, <code>xhost</code> turns on access control. This means that only the user who starts X-Windows could start PCC. You may turn off X-Windows client restrictions by typing <code>xhost +</code> at a terminal window.</td>
</tr>
<tr>
<td>Java Runtime Environment (JRE)</td>
<td>The JRE components required to run PCC are installed as part of Pervasive PSQL. PCC uses the &quot;local&quot; version of the JRE installed as part of Pervasive PSQL.</td>
</tr>
</tbody>
</table>
If you have met the requirements to run PCC and still are having difficulty running the utility, refer to the following troubleshooting guide.

Table 6  Troubleshooting Guide for Running PCC

<table>
<thead>
<tr>
<th>Troubleshooting Condition</th>
<th>Discussion</th>
</tr>
</thead>
</table>
| You receive the error "java.lang.UnsatisfiedLinkError." | This error typically occurs if you try to start PCC by double-clicking on the script file using a file browser application. Start PCC from a command prompt. This error can result if the LD_LIBRARY_PATH variable is not set. The PCC script sets this variable for you. You may also explicitly set the variable with the following command:  
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/psql/lib |
| You receive the error "SWT no more handles" when trying to run PCC as root or as user psql. | You are not required to log in as user psql or root to run PCC. However, if you are neither of these users, you must be a member of group pvsw.  
The "SWT no more handles" error is caused by X-Server denying a connection to a client. Before switching to user psql or root, open a console window and type xhost + to allow other clients to connect to X-Server.  
Now you can switch to user psql or root.  
Also, sometimes the display environment variables needs to be set. As user psql or root, type the following command at a console window:  
export DISPLAY=:0.0  
or  
export DISPLAY=localhost:0.0 |
| You want to view the error log file for PCC or redirect the errors to the console window. | By default, the log file of PCC errors is located in a subdirectory of the user’s home directory (the subdirectory is dir_pcc/workspace/.metadata). For troubleshooting, you may find it more convenient to redirect the errors to the console window.  
To redirect errors to the console window, use the "-consoleLog" option when starting PCC:  
pcc -consoleLog |
Using Pervasive PSQL Control Center

Table 6  Troubleshooting Guide for Running PCC

<table>
<thead>
<tr>
<th>Troubleshooting Condition</th>
<th>Discussion</th>
</tr>
</thead>
</table>
| You receive the following error message: "Unable to connect to database engine. Make sure the target machine is accessible and an engine is running on the target machine." | The context of this error occurs if you attempt to administer the local server. To administer the local server, you must be a member of the pvs

You receive the error: "GTK IM Module SCIM: Cannot connect to Panel!" then trying to run PCC as a user other than root. | On some Linux operating systems, it is necessary to specify the environment variable GTK_IM_MODULE. To resolve this problem, before starting PCC, run the following command at the console window:  

export GTK_IM_MODULE=scim-bridge  

PCC caches certain information to improve efficiency. The cache must be cleared after you install or upgrade any other products that "plug in" to PCC. Otherwise, the installed or upgraded product does not appear in the Pervasive PSQL Explorer. For example, if you were to install or upgrade Pervasive Data Exchange, you would then need to clear the PCC cache.

The cache can be cleared only by starting PCC with a parameter from the command line.

To clear PCC cache

1. Exit PCC if it is running (click **File** ➔ **Exit**).

2. Open a command prompt, or terminal window, at the operating system.

3. Change directory to the PSQL\bin\ folder in the Pervasive PSQL installation directory.

   For default locations of Pervasive SQL files, see Where are the Pervasive SQL files installed? in Getting Started With Pervasive SQL.

4. Type *pcc -clean* and press **Enter**.

   This starts PCC and clears the cache. The newly installed or upgraded products should then appear in the Pervasive PSQL Explorer.
An Overview of Pervasive PSQL Control Center

Note Use of the -clean parameter when starting PCC provides no advantage under normal usage. The parameter is required only if you install or upgrade a plug-in product.

Editors and Views Within PCC

The PCC main window comprises editors and views:

- Pervasive PSQL Explorer
- SQL Editor
- Grid
- Text
- Outline
- Table Editor

You can display and work with objects through the different editors and views. Multiple editors of the same type (for example, SQL Editor) can be open at the same time. Each object being edited is represented by a tab on top of the editor. The tab contains the name of the object. Data modified within an editor must be explicitly saved (for example, with File > Save).

Views, such as Pervasive PSQL Explorer, can be opened only one at a time. Actions performed within a view are applied immediately. No explicit save is required.

Editor and View Characteristics

The following table summarizes the characteristics of the editors and views.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Pervasive PSQL Explorer</th>
<th>SQL Editor</th>
<th>Grid</th>
<th>Text</th>
<th>Outline</th>
<th>Table Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contains icons specific to that view</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can be closed</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Can be minimized</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Can be maximized</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

61
Using Pervasive PSQL Control Center

Table 7  Characteristics of PCC Window Views

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Pervasive PSQL Explorer</th>
<th>SQL Editor</th>
<th>Grid</th>
<th>Text</th>
<th>Outline</th>
<th>Table Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can be restored to previous size</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Can be opened in new window</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Can be rearranged within PCC main window</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Can be &quot;pulled&quot; from PCC and placed on desktop</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

**Pervasive PSQL Explorer**

This view displays a tree of objects that can be opened or expanded to reveal more detail. Click the expand icon to the left of an object to reveal subordinate objects. The expand icon may be a “+” symbol, a “−” symbol, or some other similar symbol. Click the collapse icon to hide subordinate objects. (The collapse icon appears after you click an expand icon.)

The tree of objects includes a root node named Pervasive PSQL. The root node contains subordinate objects such as clients, services (if applicable), engines, databases, tables, views, stored procedures, user-defined functions, triggers, groups, users, and system objects (such as system tables).
Accessing Object Properties

A right-click on an object reveals actions or dialogs applicable to that object. For example, you can right-click on an object then click Properties to display configurable settings for the object (provided properties apply to the object). You may also click an object then press **Alt+Enter** to display properties.

SQL Editor

SQL Editor allows you to run Structured Query Language (SQL) statements against a Pervasive PSQL database. See SQL Editor for a detailed discussion.

Grid

The Grid window view shows in a matrix format, like a spreadsheet, the result of running SQL statements. Each field is represented as a column and the data appears in cells within the columns. You can change data directly in the Grid cells as well as add additional rows to the Grid.

Both Table Editor and SQL Editor use the Grid. See To view table data and Grid Window View for further details.
Using Pervasive PSQL Control Center

Text
The Text window view shows in a text format the result of running SQL statements. The text is display only. You cannot change data values by changing the text, but you can copy text. See Text Window View for a detailed discussion.

Outline
The Outline window view allows you to view the SQL statements in a tree structure. The root node of the tree is the same name as the name of the SQL Editor window view. See Outline Window View for a detailed discussion.

Note that the editor must support an outline or the Outline window view is not available. Currently, only SQL Editor supports an outline view.

Table Editor
Table Editor allows you to add, delete, or change the characteristics of columns within a table. The table may be one newly created or an existing table that you want to edit. See Table Editor for a detailed discussion.

Preferences
You can set general preferences for your experience in PCC. You can also set preferences for the window views in PCC or for the external tools.

➢ To set general preferences for Grid
1. On the PCC Window menu, click Preferences. Expand the Pervasive node if it is not already expanded.
2. Click General.

The following are the options that can be set in the General Preferences:

- Always remove associated DSN entries (see Deleting DSNs)
- Do not prompt for new database each time a SQL document is opened (see To set database context for an SQL query)

Select Always remove associated DSN entries to have all DSN entries for any database automatically deleted along with the database without prompting.
An Overview of Pervasive PSQL Control Center

Unselect **Do not prompt for new database each time a SQL document is opened** to be prompted to select a database each time you open a SQL document in the SQL Editor. If this option is unselected, select it to use the most recently selected database when you open a SQL document. The selected database is not maintained across PCC sessions. If you close and reopen PCC, you will have to select a new default database context.

**Preferences for PCC Window Views**

You can set preferences for the following PCC window views:

- Grid
- Monitor
- SQL Editor
- Table Editor
- Text

➢ **To set preferences for PCC Window Views**

1. On the PCC Window menu, click **Preferences**. Expand the Pervasive node if it is not already expanded.

2. Perform one of the following actions:
   a. To set preferences for Data Grid, click **Data Grid**.
   b. To set preferences for Monitor, click **Monitor**.
   c. To set preferences for SQL Editor, click **SQL Editor**.
   d. To set preferences for Table Editor, click **Table Editor**.
   e. To set preferences for Text Output, click **Text Output**.
**Additional Utilities**

Some utilities have not yet been tightly integrated within the PCC framework. However, they may still be started from within PCC by selecting them through the **Tools** menu:

- Function Executor (see *Testing Btrieve Operations* in Advanced Operations Guide)
- Monitor (see *Monitoring Database Resources* in Advanced Operations Guide)
- Maintenance (see *Manipulating Btrieve Data Files with Maintenance* in Advanced Operations Guide)
- DDF Builder (see *Getting Started with DDF Builder* in DDF Builder User’s Guide)
- License Administrator (see License Administration)
- Rebuild (see *Converting Data Files* in Advanced Operations Guide)
- ODBC Administrator — separate choices for 32-bit Administrator and 64-bit Administrator on 64-bit operating systems (see *DSNs and ODBC Administrator* in SQL Engine Reference). Note that Windows uses whichever ODBC Administrator is open if you try to invoke the other one. That is, if the 32-bit ODBC Administrator is open and you attempt to start the 64-bit one, Windows displays the 32-bit version (and vice versa). In other words, only one version of ODBC Administrator runs at a time. This is a limitation of Windows, not Pervasive PSQL.
- Pervasive System Analyzer (see Pervasive System Analyzer (PSA))
- Query Plan Viewer (see Query Plan Viewer in SQL Engine Reference)
- Gateway Locator (if PCC installed with Pervasive PSQL Workgroup) (see Gateway Configuration in Getting Started With Pervasive PSQL)

**Note** These utilities appear in the Tools menu only on Windows platforms. For both Windows and Linux platforms, you can add your own custom tools to the Tools menu. Refer to the next section.
External Tools

You can add your own software programs to the PCC Tools menu. This provides a convenient way to start the programs from PCC.

▶ To add external tools

1. On the PCC Window menu, click Preferences. Expand the Pervasive node if it is not already expanded.
2. Click External Tools.
3. Click New.
4. Type a name for Tool Label that you want to appear in the Tools menu.
5. Type the path and file name of the program for Tool Location. You can click … and browse to the file location if your prefer.
6. Optionally, type any parameters for Tool Parameters that should be passed to the program when the program starts.
7. Click OK.
8. Click OK (or Apply then OK) to close the Preferences dialog.

▶ To set preferences for external tools

1. On the PCC Window menu, click Preferences. Expand the Pervasive node if it is not already expanded.
2. Click External Tools.
3. Click the desired tool in the External Tools list.
4. Perform one of the following actions:
   a. To remove the tool from the list, click Remove.
   b. To move the tool toward the top of the list, click Up.
   c. To move the tool toward the bottom of the list, click Down.
Services on Windows Servers

PCC offers a convenient way to work with Pervasive PSQL servers on Windows machines without having to use the Windows Services control panel. You can start, stop, and set the startup policy from within PCC.

You must stop the relational and transactional services to completely stop Pervasive PSQL. Stopping just one of the services does not stop the database engine completely.

This section applies only to Windows platforms, not to Linux.

Note Pervasive PSQL products other than the database engine also run as services. The services of these other products can have a dependency on the database engine services. See Services Dependencies for further information.

To start or stop services

1. In Pervasive PSQL Explorer, expand the Services node in the tree.

2. Right-click on the service that you want to stop or start.

3. Perform one of the following actions:
   a. Click Start Service to start the service
   b. Click Stop Service to stop the service
   c. Click Restart Service to stop then start the service.

Tip You can also stop or restart all services with a single command. Right-click the Services node in the tree, then click Stop All Services or Restart All Services.
To set services startup policy

1. In Pervasive PSQL Explorer, expand the Services node in the tree.
2. Right-click on the service for which you want to set a startup policy.
3. Click Properties.
4. Click the desired policy:

<table>
<thead>
<tr>
<th>Startup Policy</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>You must manually start the service after the operating system starts.</td>
</tr>
<tr>
<td>Automatic</td>
<td>The service automatically starts when the operating system starts.</td>
</tr>
<tr>
<td>Disabled</td>
<td>The service is removed from operation and is not affected by starting the operating system.</td>
</tr>
</tbody>
</table>

5. Click OK (or Apply then OK).

See To set services startup policy by using PCC in Advanced Operations Guide.
You can use PCC to work with database engines that are on your machine or with remote server engines. To work with a remote server engine, you must introduce it to PCC. This procedure is called registering the server.

Your local server is automatically registered to PCC when you install Pervasive PSQL v11 SP3. The local server appears in the Pervasive PSQL Explorer as the first entry under the Engines node.

To register a remote server engine

1. In Pervasive PSQL Explorer, right-click on the top node in the tree (Pervasive PSQL).
2. Click New > Server.
3. Identify the server that you want to register.
   - Type the name by which the server is identified on the network or type the IP address of the server.
4. Click Finish.
   - The server should now appear in the Pervasive PSQL Explorer window of PCC under the Engines node.

To log out from a database engine

These steps do not erase the data or database from the server. A logout only disconnects the communication between the database engine and the PCC on your computer.

1. In Pervasive PSQL Explorer, expand the Engines node.
2. Right-click on the server engine from which you want to log out.
3. Click Logout (name).
   - Name reflects the name of the user currently logged in to the server through PCC. Name is anonymous if no specific user name and password were provided for a login.
   - Any nodes expanded for the database engine are collapsed.
To reconnect to a database engine

1. In Pervasive PSQL Explorer, expand the node in the tree for the server engine.

To log in to a database engine

1. Right-click on the database name in the PCC Pervasive PSQL Explorer then click Logout (name).

   Name reflects the name of the user currently logged in to the server through PCC. By default, name is **anonymous**, meaning that no specific user name and password were provided for a login.

   Any nodes expanded for the database engine are collapsed.

2. Right-click on the database name.

3. Click Login.

4. Type a **User Name** and **Password**.

   You can leave these blank to log in as anonymous.

5. Click **OK**.

See **Configuration**.
Capacity Usage Viewer

PCC provides Capacity Usage Viewer to monitor concurrent sessions and data usage for all database engines. This is especially useful when you are considering migrating from PSQL Server to PSQL Vx Server, because of the difference in the way those two editions are licensed. PSQL Server licenses are based on the number of users, while PSQL Vx Server licenses are based on the number of concurrent sessions and the amount of data in use.

Capacity Usage Viewer includes two graphs, one for the number of concurrent sessions and one for the amount of data. Each graph includes a usage level bar, a heavy horizontal line across the graph, to help you determine what volume of usage is normal and what volume is uncommon for your business. The Capacity Usage Viewer also displays peak usage statistics.

The graphs use the peak values that are recorded each day. For any day on which the engine is not used, they use a value of zero. They require a minimum of two days’ data to be generated. Otherwise, Capacity Usage Viewer displays an error message.

➤ To access the Capacity Usage Viewer

In PCC Pervasive PSQL Explorer, right-click the engine you want to examine.

Its context menu opens.

1 Click Capacity Usage Graphs... from the context menu.

Capacity Usage Viewer GUI

The following image shows the GUI. The table below the image describes the GUI objects. Click on an area of the image for which you want more information.
### GUI Object | Description
--- | ---
Title Bar | Identifies the engine you selected.
Statistical Indicators | Displays meaningful statistics and enables you to select what you want to display in the graphs. See Statistical Indicators.
Time Designation | Displays the starting and ending dates of the period for the data displayed in the graphs.
Session Usage Graph | Displays, graphically, the number of sessions that occurred concurrently during the selected time period.
Data Usage Graph | Displays, graphically, the amount of data used during the selected time period.
Usage Level Bars | Enables you to determine how often your usage exceeds a selected level. When selected, they appear across each graph at a default level that is 90% of peak usage. The number at the left of each usage level bar identifies its level (amount of data or number of sessions). You can move the usage level bar to whatever level you need, either by using the spin boxes or by dragging with the cursor. The two usage level bars are independent of each other.
Zoom Instructions | Describes the general procedures for zooming in and out of a graph. For detailed procedures, see Zooming.
Export Button | Enables you to export the data to a .CSV file, if you find it useful to save the data for additional analysis. The Export button opens a Browse for Folder dialog, where you can select a location for data storage.
Using Pervasive PSQL Control Center

**Figure 21  Statistical Indicators**

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Period**       | Enables you to select the time period for the data you want the graphs to display. When the window opens, the graphs display, by default, the data from the period that was selected when the window was last closed. You can select a different time period:  
  1. All  
  2. Last week  
  3. Last 30 days  
  4. Last 90 days  
  5. Last 180 days  
  You can also select a time period by zooming the graphs. When you zoom a graph, the **Period** drop-down menu displays **Custom** as the selected time period. |
| **Peak Statistics Group Box** | Contains fields that display statistics for maximum use of data and maximum number of concurrent sessions during the time period displayed in the graphs. |
| **Peak Sessions** | Displays the greatest number of concurrent sessions that occurred during the time period displayed in the **Session Usage** graph. |
| **Peak Sessions Date** | Displays the date on which the greatest number of concurrent sessions occurred. If that number of sessions occurred more on more than one day, the most recent date is displayed. |
Capacity Usage Viewer

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak Data (GB)</td>
<td>Displays the maximum amount of data, in gigabytes, used at one time during the time period displayed in the Data Usage graph.</td>
</tr>
<tr>
<td>Peak Data Date</td>
<td>Displays the date on which the maximum amount of data was used. If that amount of data was used on more than one day, the most recent date is displayed.</td>
</tr>
<tr>
<td>Show Usage Level Bars</td>
<td>Displays or hides usage level bars across the graphs, depending on whether it is checked or not.</td>
</tr>
<tr>
<td>Usage Statistics Group Box</td>
<td>Contains spin boxes for moving the usage level bars up and down and fields that display the statistics that result from moving the usage level bars.</td>
</tr>
<tr>
<td>Sessions</td>
<td>Sets the value at which to position the usage level bar in the Session Usage graph.</td>
</tr>
<tr>
<td>Days Exceeded</td>
<td>Displays the number of days on which the number of concurrent session in use was greater than the level at which the usage level bar is set.</td>
</tr>
<tr>
<td>Data (GB)</td>
<td>Sets the value at which to position the usage level bar in the Data Usage graph.</td>
</tr>
<tr>
<td>Days Exceeded</td>
<td>Displays the number of days on which the amount of data in use was greater than the level at which the usage level bar is set.</td>
</tr>
</tbody>
</table>

**Zooming**

If you need to view a particular time period besides the selections provided in the Period pull-down menu, you can select and zoom a segment of a graph. When you zoom one graph, the other zooms simultaneously. The two graphs are always set to the same period.

- **To zoom a graph**
  1. Place the cursor at the start of the period you want to display.
Using Pervasive PSQL Control Center

2 With your left mouse button pressed down, drag the cursor to the end of the period you want to display. In the graph, a rectangular black outline appears around the selected part of the graph. As you move the cursor, the rectangle enlarges accordingly.

3 Release the mouse button. The graph is re-rendered to display the period you selected. The Period field displays the setting Custom.

4 Repeat the process to zoom further.

5 To zoom out, place the cursor anywhere in the graph and, with your left mouse button pressed down, drag the cursor from right to left. The graph returns to its original period, that is, the period it displayed when the window opened.
Monitor

PCC integrates a Monitor utility which allows you to monitor certain activities and attributes of the database engine. The utility can monitor aspects of the transactional interface and the relational interface. It presents information organized into a series of tabs, that can be rearranged for your convenience, with columns of data that can rearranged and sorted. It presents a snapshot of a particular moment and can be refreshed either manually or automatically.

Accessing Monitor

Within PCC, you access Monitor from Pervasive PSQL Explorer.

➢ To access Monitor for a selected database engine

1. In Pervasive PSQL Explorer, locate and open the Engines node.
2. Right-click the database engine that you want to monitor. Its context menu opens.
3. Click Monitor Server on the context menu.
   You can monitor multiple engines at the same time if you choose.

Features of the Monitor User Interface

The table below the following image explains the user interface components. Click any area of the image for which you want more information.
Using Pervasive PSQL Control Center

Figure 22  Monitor User Interface

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>File menu</td>
<td>Allows you to perform the following commands:</td>
<td>Setting Refresh Options</td>
</tr>
<tr>
<td></td>
<td>• Automatic Refresh</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Set Automatic Refresh Rate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Close Monitor Interface</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Exit from Pervasive Control Center</td>
<td></td>
</tr>
<tr>
<td>Window menu</td>
<td>Allows you to display the following PCC and Monitor elements:</td>
<td>Setting Monitor Preferences</td>
</tr>
<tr>
<td></td>
<td>• Database window of the PCC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Any individual Monitor tab</td>
<td></td>
</tr>
<tr>
<td></td>
<td>You can also use this menu to set preferences for the Monitor UI and other</td>
<td></td>
</tr>
<tr>
<td></td>
<td>elements of the PCC.</td>
<td></td>
</tr>
<tr>
<td>Help menu</td>
<td>Allows you to access the following information:</td>
<td>Pervasive PSQL Event Log in Pervasive PSQL User's Guide</td>
</tr>
<tr>
<td></td>
<td>• PSQL Documentation Library</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• PVSW Log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• PCC Log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• PCC version and copyright information</td>
<td></td>
</tr>
<tr>
<td></td>
<td>You can also use this menu to clear the PCC log.</td>
<td></td>
</tr>
</tbody>
</table>
You can set preferences for the Monitor from either Monitor itself or from PCC. In either utility, select Window > Preferences > Monitor to open the Monitor tab of the Preferences dialog box.

There are two types of preferences you can set. First, you can set the layout of the Monitor user interface, so that when you reopen Monitor, its tabs are arranged just as you left them. You can set this individually for each server that you access. Second, you can set the features for any particular grid, for example, column width, sort order, and column order. If you open that particular grid on another server, it follows the same settings. That way, you can easily compare the same grid on different servers.

Information in the Monitor can be refreshed automatically at a configured interval, as desired, or not at all. Be aware that refreshing too many windows at a short interval may slow performance.

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Automatic Refresh</td>
<td>Toggles between automatic refresh on and automatic refresh off</td>
<td>Setting Refresh Options</td>
</tr>
<tr>
<td>Set Refresh Rate</td>
<td>Displays a dialog box in which you can set the number of seconds between each repopulation of the user interface with current information</td>
<td>Setting Refresh Options</td>
</tr>
<tr>
<td>Tab Set area</td>
<td>Includes five tabs with grids that display information about activity occurring at that moment in the database. When you select a tab, a selection of icons that apply to that tab appear at the right of the row of tab labels.</td>
<td>Tab Functionality Monitor Server Activities</td>
</tr>
</tbody>
</table>

**Setting Monitor Preferences**

**Setting Refresh Options**
Using Pervasive PSQL Control Center

**Tab Functionality**

Tabs can be rearranged, separated, and reaggregated for your convenience. To move a tab, put the cursor on the tab label, hold down the left mouse button, and pull the tab label where you want the tab to be.

Because of the different nature of the data on each tab, different operations can be performed on each tab. Those operations are initiated by the icons that appear at the right end of the row of tab labels. The following table describes the icons and lists the tabs on which they appear.

Table 8  Tab Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
<th>Tabs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh</td>
<td>Repopulates the user interface with the current information.</td>
<td>All</td>
</tr>
<tr>
<td>Minimize</td>
<td>Toggles between minimize and restore. When you minimize the interface, it is reduced to a series of icons at the right of the screen, one for each tab and the restore icon.</td>
<td>All</td>
</tr>
<tr>
<td>Maximize</td>
<td>Toggles between maximize and restore. When you maximize a selected tab in an interface with several open tabs, the maximized tab expands to fill the interface.</td>
<td>All</td>
</tr>
<tr>
<td>Select Columns to Display</td>
<td>Displays a dialog box in which you can select which columns to display. Because each tab includes different columns, each dialog box includes a different list of column names. All dialog boxes include the choices Select All and Deselect All.</td>
<td>All</td>
</tr>
<tr>
<td>Hide/Show Handles</td>
<td>Toggles between displaying or hiding the Handle Information grid at the bottom of the tab. The Handle Information grid includes different information, depending on the tab. The Handle Information grid also has a Select Columns to Display icon, so that you can hide or display columns as desired.</td>
<td>Active Files MicroKernel Sessions</td>
</tr>
</tbody>
</table>
Monitor

The following sections describe the activities displayed on the Monitor tabs:

- **Active Files**
- **MicroKernel Sessions**
- **Resource Usage**
- **MicroKernel Communication Statistics**
- **SQL Active Sessions**

### Active Files

This tab provides information about MicroKernel files that are currently open.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
<th>Tabs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete Selected Session</td>
<td>Removes a highlighted session from the tab. This icon becomes activated only after a session is selected. <strong>Caution:</strong> This procedure actually terminates a session, so you are interrupting someone’s work in progress. Consequently, a message asks you to confirm that you actually want to do this.</td>
<td>MicroKernel Sessions SQL Sessions</td>
</tr>
<tr>
<td>Delete All Sessions</td>
<td>Removes all sessions from the tab. <strong>Caution:</strong> This procedure actually terminates all sessions, so you are interrupting work in progress. Consequently, a message asks you to confirm that you actually want to do this.</td>
<td>MicroKernel Sessions</td>
</tr>
<tr>
<td>Reset Deltas</td>
<td>Deletes a statistic and restarts the count at zero.</td>
<td>MicroKernel Communication Statistics</td>
</tr>
</tbody>
</table>

### Table 8  Tab Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
<th>Tabs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Provides the directory and all subdirectories to the location of the file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Indicates the name, including suffix, of the file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Indicates the size in bytes of each page in the file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Indicates whether the file is flagged as read-only by the operating system.</td>
<td></td>
</tr>
</tbody>
</table>
Using Pervasive PSQL Control Center

### Record Locks
Indicates whether any of the active handles for the selected file have record locks. Any application can read a locked record, but only the application that placed the lock can modify or delete the record. A record lock exists only as long as the application that opened the file is updating a record. "Yes" indicates that one or more record locks are applied to the file. "No" indicates that no records are locked.

### Transaction Lock
Indicates whether any of the active handles for the selected file have a transaction lock. A transactional file lock exists only as long as the application that opened the file is processing a transaction.

### Physical File Size
Indicates the size of the file in kilobytes (KB). This information is particularly useful for the capacity-based license model if you want to review data in use on a file-by-file basis. See also Capacity-based License Model in Pervasive PSQL User's Guide.

Monitor uses kilobytes (KB) for the size of an individual file and megabytes (MB) as the units for resource usage (Viewing Resource Usage). License Administrator uses gigabytes (GB) as the units because that is how data in use is associated with a key. The different contexts require units appropriate for each context.

If a file is immediately closed after you insert a large number of records, Monitor does not immediately reflect the changes in file size. For example, the statistics for “Physical File Size KB” are not refreshed for that file until the next time the file is opened for reading or writing.

You can view the handle information for any active file, if the Hide/Show Handles icon is set to display handles. Active file handles include the following information.

<table>
<thead>
<tr>
<th>Client</th>
<th>Indicates the name (typically the login ID of the user) or an index to the Client list of the database server.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Number</td>
<td>Displays the network connection number of the client. If the client does not have a network connection, this field displays “NA” for “not applicable”</td>
</tr>
<tr>
<td>Task Number</td>
<td>Displays the process-supplied task number for processes originating at the server or a Windows Client.</td>
</tr>
<tr>
<td>Site</td>
<td>Specifies the location of the user process (local or remote).</td>
</tr>
</tbody>
</table>
Monitor

**Network Address**
Identifies the location of the calling process on the network. If the calling process is SPX, the network node/network address is preceded by S: such as $S: 65666768 0000000001$.

If the calling process is TCP/IP, the address is preceded by T4 for IPv4 addresses, T6 for IPv6 addresses, and T for the fully qualified domain name of a client machine. Examples:
- T4: 180.150.1.24
- T6: 1234:5678:0000:0000:0000:0000:9abc:def0
- T: `<mymachine.mydomain.mycompany>.com`

**Open Mode**
Indicates the method the application uses to open the specified handle of the file. Valid open modes are the following:
- Normal – The application that opened the file has normal shared, read/write access to it.
- Accelerated – The application that opened the file has shared read/write access.
- Read-only – The application that opened the file has read-only access; it cannot modify the file.
- Exclusive – The application that opened the file has exclusive access. Other applications cannot open the file until the calling application closes it.

Monitor also specifies all open modes as *non-transactional or shared locking* when applicable.

**Record Lock Type**
Displays the type of record lock(s) currently held by the handle. The possible record lock types are Single, Multiple, and None.

Single-record locks enable a user to lock only one record at a time. Multiple-record locks enable a user to lock more than one record at a time.

**Wait State**
Indicates whether the user is waiting because of some type of lock on this handle: Waits for Record Lock, Waits for File Lock, or None.

**Transaction State**
Displays the state of the transaction lock currently held by the handle. The possible transaction types are Exclusive, Concurrent, or None.

### MicroKernel Sessions

This tab provides information about current connections to the storage engine.

<table>
<thead>
<tr>
<th>Session</th>
<th>Provides a unique identifier for the connection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Number</td>
<td>Displays the network connection number of the session. If the session does not have a network connection, this field displays “NA” for “not applicable”.</td>
</tr>
</tbody>
</table>
### Using Pervasive PSQL Control Center

<table>
<thead>
<tr>
<th>Task Number</th>
<th>Displays the process-supplied task number for processes originating at the server, or from a Windows Client.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site</td>
<td>Specifies the location of the session process (local or remote).</td>
</tr>
<tr>
<td>Network Address</td>
<td>Identifies the location of the calling process on the network. If the calling process is SPX, the network node/network address is preceded by S: such as S: 65666768 0000000001. If the calling process is TCP/IP, the address is preceded by T4 for IPv4 addresses, T6 for IPv6 addresses, and T for the fully qualified domain name of a client machine. Examples: T4: 180.150.1.24 T6: 1234:5678:0000:0000:0000:9abc:def0 T: &lt;mymachine.mydomain.mycompany&gt;.com If multiple clients from a single machine connect by different TCP/IP addresses, each address is valid for that client. However, internally to the database engine, an address associated with a client may not be the actual address used by that client. This is because of the way the database engine identifies and manages multiple clients from the same machine. Consequently, since Monitor is reporting engine information, the utility may display an associated address instead of the actual address.</td>
</tr>
<tr>
<td>Locks Used</td>
<td>Indicates the number of locks the session is currently using.</td>
</tr>
<tr>
<td>Transaction State</td>
<td>Displays the type of transaction lock the session currently holds. The possible transaction types are Exclusive, Concurrent, or None.</td>
</tr>
<tr>
<td>Records Read</td>
<td>Displays the number of records read since the session first opened a file.</td>
</tr>
<tr>
<td>Records Inserted</td>
<td>Displays the number of records the session has inserted.</td>
</tr>
<tr>
<td>Records Deleted</td>
<td>Displays the number of records the session has deleted.</td>
</tr>
<tr>
<td>Records Updated</td>
<td>Displays the number of records the session has updated.</td>
</tr>
<tr>
<td>Disk Access</td>
<td>Indicates the number of times the session required a disk access. You will not see any information for disk accesses for files that have just been opened.</td>
</tr>
<tr>
<td>Cache Access</td>
<td>Indicates the number of times this client experiences a miss of the L1 cache and moves a page from either L2 cache or the disk into the L1 cache in order to fulfill the request.</td>
</tr>
</tbody>
</table>
You can view the handle information for any MicroKernel session, if the Hide/Show Handles icon is set to display handles. MicroKernel session handles include the following information.

<table>
<thead>
<tr>
<th>Path</th>
<th>Provides the directory and all subdirectories to the location of the file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>Indicates the name, including suffix, of the file.</td>
</tr>
<tr>
<td>Open Mode</td>
<td>Indicates the method the application uses to open the specified handle of the file. Valid open modes are:</td>
</tr>
<tr>
<td></td>
<td>Normal – The application that opened the file has normal shared, read/write access to it.</td>
</tr>
<tr>
<td></td>
<td>Accelerated – The application that opened the file has shared read/write access.</td>
</tr>
<tr>
<td></td>
<td>Read-only – The application that opened the file has read-only access; it cannot modify the file.</td>
</tr>
<tr>
<td></td>
<td>Exclusive – The application that opened the file has exclusive access. Other applications cannot open the file until the calling application closes it.</td>
</tr>
<tr>
<td></td>
<td>Monitor also specifies all open modes as non-transactional or shared locking when applicable.</td>
</tr>
<tr>
<td>Record Lock Type</td>
<td>Displays the type of record lock(s) currently held by the handle. The possible record lock types are Single, Multiple, and None.</td>
</tr>
<tr>
<td>Wait State</td>
<td>Indicates whether the user is waiting because of some type of lock on this handle: Waits for Record Lock, Waits for File Lock, or None.</td>
</tr>
<tr>
<td>Transaction State</td>
<td>Displays the state of the transaction lock currently held by the handle. The possible transaction types are Exclusive, Concurrent, or None.</td>
</tr>
</tbody>
</table>

**Resource Usage**

This tab shows values for different aspects of MicroKernel operations in progress. In addition to the current values (defined in the following table), the grid on the tab provides peak and maximum values, enabling you to assess resource use and needs. “Peak” is the highest value for the resource since the MicroKernel was started.
"Maximum" is the highest allowed value. The following table indicates which resources have unlimited maximum values.

<table>
<thead>
<tr>
<th>Files</th>
<th>Indicates the number of files currently open by the MicroKernel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handles</td>
<td>Indicates the number of active handles. The MicroKernel creates a handle each time a user opens a file. A single session can have several handles for the same file.</td>
</tr>
<tr>
<td>Clients</td>
<td>Indicates the number of clients accessing the MicroKernel. A machine can have multiple clients accessing the database engine simultaneously. The engine dynamically manages the client list. The number of clients is limited only by the memory in the computer. “Client” indicates a session established by a client ID (transactional engine interface) or a connection to the relational engine interface. The database engine uses various client sessions for its own internal processes, such as for accessing Pervasive PSQL system files, metadata files, dbnames.cfg, and default system databases. The number of clients indicates both internal client sessions and non-internal client sessions (see Viewing Session Information).</td>
</tr>
<tr>
<td>Worker Threads</td>
<td>Indicates the number of concurrent MicroKernel processes.</td>
</tr>
<tr>
<td>User Count</td>
<td>Indicates the number of concurrently connected users. The maximum value shows the maximum permitted users as granted by a license agreement.</td>
</tr>
</tbody>
</table>
| Session Count  | Indicates the number of sessions in use by the database engine. For brevity, “number of sessions in use” is also referred to “session count.” The maximum value (also called the “session count limit”) shows the maximum permitted sessions as granted by a license agreement.

Session count reflects all sessions whether established through the transactional interface or through the relational interface.

Messages pertaining to session count are logged to the various Pervasive PSQL logging repositories. See Pervasive PSQL Message Logging in Pervasive PSQL User’s Guide.

The database engine uses various sessions for its own internal processes, such as for accessing Pervasive PSQL system files, metadata files, dbnames.cfg, and default system databases. These internal sessions do not consume any session counts.
Monitor

MicroKernal Communication Statistics

This tab displays information about communication with the data storage engine. It includes separate sections for communication statistics and resource usage information.

Communication statistics are calculated in terms of total number of occurrences processed since the database engine was started. “Delta” indicates the number of occurrences since you last accessed the

| Data In Use MB | Indicates in megabytes (MB) the size of all concurrently open data files. The maximum value is the maximum permitted amount of all concurrently open data files as granted by a license agreement. The maximum is also called the “data in use limit.”

The value for data in use increases when a data file is first opened. Subsequent opens to an already open data file do not add to the total. Data in use also increases if an open file increases in size. Operations on an already open file continue to be permitted even if the size of the open file increases beyond the data in use limit.

The value for data in use decreases when a data file is closed by the final user to have the file open. Since more than one user can access the same data file, all opens must be closed before data in use decreases.

Messages pertaining to data are logged to the various Pervasive PSQL logging repositories. See Pervasive PSQL Message Logging in Pervasive PSQL User's Guide.

The database engine uses various files for its own internal processes, such as Pervasive PSQL system files, metadata files, dbnames.cfg, and default databases. Files used for internal processes do not increase the value for data in use.

If a file is immediately closed after you insert a large number of records, Monitor does not immediately reflect the changes in file size. For example, the statistics for “Data in Use MB” are not refreshed for that file until the next time the file is opened for reading or writing.

Transactions | Indicates the number of transactions. The maximum for this resource is unlimited.

Locks | Indicates the number of record locks. The maximum for this resource is unlimited.
Using Pervasive PSQL Control Center

MicroKernel **Communications Statistics** tab. To restart the count of the delta number, click **Reset Deltas**.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Requests Processed</td>
<td>Indicates the number of requests the database engine has handled from workstations or remote, server-based applications.</td>
</tr>
<tr>
<td>SPX Requests Processed</td>
<td>Indicates the number of SPX requests the database engine has handled from clients or remote, server-based applications.</td>
</tr>
<tr>
<td>TCP/IP Requests Processed</td>
<td>Indicates the number of TCP/IP requests the database engine has handled from clients or remote, server-based applications.</td>
</tr>
<tr>
<td>NetBIOS Requests Processed</td>
<td>Indicates the number of NetBIOS requests the database engine has handled from clients or remote, server-based applications.</td>
</tr>
<tr>
<td>Connection Timeouts</td>
<td>Indicates the number of times Auto Reconnect has timed out when attempting to reconnect to Clients. See also <strong>Auto Reconnect Timeout</strong>.</td>
</tr>
<tr>
<td>Connection Recoveries</td>
<td>Indicates the number of times the AutoReconnect feature has successfully recovered from a connection timeout.</td>
</tr>
</tbody>
</table>

Resource usage information provides current values, that is, number of occurrences. In addition, for comparison purposes, it provides a peak usage number (the highest number of occurrences for the resource since the MicroKernel was started) and a maximum allowed value. The following table defines the current value for each resource and explains the maximum value when necessary.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communications Threads</td>
<td>Indicates the number of remote requests that the MicroKernel is currently processing. Local requests are not included in this statistic. For the total number of remote and local threads being processed, see <a href="#">Viewing Resource Usage</a>. The database engine dynamically increases the number of communications threads as needed up to the maximum allowed. For Windows and Linux platforms, the maximum is 1,024. Worker threads are also used to process Monitor requests, so you might not see the number of current worker threads drop below one. This is normal.</td>
</tr>
<tr>
<td>Total Remote Sessions</td>
<td>Indicates the number of remote clients connected to the database engine. The maximum number is dynamic and displays as zero.</td>
</tr>
</tbody>
</table>
### Monitor

| SPX Remote Sessions | Indicates the number of remote clients connected through the SPX protocol to the database engine. |
| TCP/IP Remote Sessions | Indicates the number of remote clients connected through the TCP/IP protocol to the database engine. |
| NetBIOS Remote Sessions | Indicates the number of remote clients connected through the NetBIOS protocol to the database engine. |

## SQL Active Sessions

This tab provides information about current connections to the relational interface.

| User Name | Provides the login name of the user. |
| Client Host Name | Identifies the name of the Client machine for the selected User Name. If unavailable, this is set to “Unknown.” |
| Network Address | Identifies the Client machine’s IP or SPX address for the selected User Name. If unavailable, this is set to “Unknown.” Values displayed include IP, SPX, Shared Memory and Unknown. |
| Client Application | Identifies the connected application or module. If unavailable, this is set to “Unknown.” |
| Data Source Name | Identifies the name of the DSN referenced by the Client application. |
| Connection Status | Specifies the connection status for the selected User Name. A status can be any of the following:  
  - Active – The session has files open. and that Idle means that the session has no files open.  
  - Idle – The session has no files open.  
  - Dying – A temporary status that indicates an active session has been deleted but has not finished processing the SQL code. At a suitable termination point, the session is no longer listed on the SQL Active Session dialog.  
  - Unknown – Status is unavailable. |
| Active/Idle Period | Displays the duration of time, in milliseconds, since the connection has been active or idle. |
| Total ConnectionTime | Displays the duration of time, in seconds, since the connection has been established. |
Databases

A database is a collection of data stored together. Newly created databases are empty and may be populated with tables. Refer to the chapter Pervasive PSQL Databases in Advanced Operations Guide for a detailed discussion of databases.

The properties of a database include such items as file locations, referential constraints, security, and whether the database is bound.

Note: If you wish to add a database to a Server engine, you must have administrative rights on the server operating system. If you do not have administrative rights, you will not be permitted to add the database.

To log out from a database

1. In Pervasive PSQL Explorer, expand the Engines node.
2. Expand the node for the registered server to display the databases on that server.
3. Right-click on the database from which you wish to log out.
4. Click Logout (name).

Name reflects the name of the user currently logged in to the database. If the database does not have security enabled, name is Master. Name may also be Master if the current user is logged in as Master.

Any nodes expanded for the database are collapsed.

Database Properties

You set properties for a database from a Properties dialog in PCC.

The dialog contains a tree with the following property nodes:

- Code Page
- Directories
- General
- Relational Constraints
- Security
Databases

Code Page
This section details the property settings for Code Page.

- Database Code Page
- PCC Connection Encoding

**Note** The database engine does not validate the encoding of the data and metadata that an application inserts into a database. The engine assumes that all data was entered using the encoding of the server or the client, as explained in Encoding Interaction in Getting Started With Pervasive PSQL.

Database Code Page
This property specifies the encoding to use for metadata and is stored in DBNAMES.cfg. Note that this property applies to the database, which means that it potentially affects all client applications that exchange data with that database. A compatible encoding must be established between the Pervasive PSQL database engine and a client application. See Encoding Interaction in Getting Started With Pervasive PSQL for the various ways in which this can be accomplished.

**Note** Changing the database code page does not convert existing data or metadata in the database. To avoid data corruption, ensure that the code page setting matches the current encoding of any pre-existing data or metadata in the database.

Database code page is particularly handy if you need to manually copy Pervasive PSQL DDFs to another platform with a different OS encoding and still have the metadata correctly interpreted by the database engine.

The default code page is “server default,” meaning the operating system code page on the server where the database engine is running. The link “Change code page” provides additional information about the setting and lets you select a specific code page.
**PCC Connection Encoding**

PCC is, itself, a client application to the database engine. As a client, PCC lets you specify the code page (the encoding) to use for each database session when PCC reads and inserts metadata and data. The default for an existing database is to use the encoding of the machine where PCC is running. This is the legacy behavior of PCC. The default for a new database is to use automatic translation.

The following explains the interaction between the settings for “PCC connection encoding” and “Database code page.”

<table>
<thead>
<tr>
<th>PCC Connection Encoding Set to a Specific Encoding</th>
<th>PCC Connection Encoding Set to “Automatic Translation”</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCC ignores Database Code Page and uses the encoding specified to read and insert data and metadata. (This is the legacy behavior of PCC.)</td>
<td>PCC and the database automatically establish compatible encoding. The database metadata and data are translated from the encoding specified for Database Code Page to the encoding used on the system where PCC is running.</td>
</tr>
</tbody>
</table>

**Note** “PCC connection encoding” applies only to PCC. It has no affect on other client applications.

When a database has OEM character data in it, the legacy solution was for the access method, such as ODBC using a DSN, to specify OEM/ANSI conversion. Now it is possible to set the OEM code page for the database and have the access method specify automatic translation. See also Automatic in SQL Engine Reference.

**Directories**

The property settings for Directories specify where certain types of files reside on physical storage.

- Dictionary Location
- Data Directories

**Dictionary Location**

This location specifies where the dictionary files (DDFs) reside on physical storage. This location must be on the same server to which...
you are connected (and where the database engine is running). The location must be formatted as though you are working directly at the server machine.

- For Windows operating systems, enter a path in the form `drive\path`, where `drive` is a drive letter on the server.
- For Linux, enter the standard Linux path format from root.

For example, if you are at a workstation connected to a Windows server where the database engine is running, and you want to create a new database on the C:\ drive of the server in the folder “mydata,” enter the location as “c:\mydata.” You would enter it this way even if you have a local network drive (for example, F:\) mapped to the server’s C:\ drive.

**Data Directories**

The Data Directories list specifies where the data files reside on physical storage. Add locations to the list by clicking New. Remove lets you remove locations from the list. The locations must be on the same server where the database engine is running.

Specify the location in the same manner as for the dictionary locations.

**General**

General contains the following property settings:

- **Bound Database**
- **Integrity Enforced**
- **Long Metadata (V2 metadata)**
- **Relational Constraints**

**Bound Database**

Indicates whether or not the database is bound. Binding a database prevents the DDFs or data files from being used in another database and prevents a data file from having two or more different table definitions within the same database.

For more information about bound databases, refer to Bound Database versus Integrity Enforced.
Integrity Enforced

Specifies whether integrity constraints (security, RI, and triggers) are enforced on the database. These constraints apply to Btrieve access to the data files as well as ODBC/SQL access.

See Setting Up Referential Integrity and Interactions Between Btrieve and Relational Constraints.

Long Metadata (V2 metadata)

This property is read-only and shows whether V2 metadata was specified when the database was created. If it was, the “Long Metadata” option is selected. Note that V2 metadata is also referred to as “long metadata.”

Relational Constraints

Relational Constraints displays a matrix that lists the relational constraints in effect for the database. See Interactions Between Btrieve and Relational Constraints.

Security

Security contains property settings (tabbed areas) for Database Security and Btrieve Security. See the chapter Pervasive PSQL Security for a complete discussion of security.
New Database GUI Reference

The following image shows the dialog with which you create a new database. The table below the image describes the GUI objects. (See also To create a new database.)

Click on an area of the image for which you want more information.

![Create New Database Dialog](image)

Table 9  Create New Database GUI Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Related Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Name</td>
<td>The name for the database that you want to appear in the database listing in PCC.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The database name cannot be the same as an existing data source name (DSN).</td>
<td></td>
</tr>
<tr>
<td>Location</td>
<td>This location must be on the same server to which you are connected (and where the database engine is running). Location must be formatted as though you are working directly at the server machine.</td>
<td>Dictionary Location and Data Directory (back to Figure 23)</td>
</tr>
<tr>
<td>Bound</td>
<td>Indicates whether or not the database is bound. Binding a database prevents the DDFs or data files from being used in another database and prevents a data file from having two or more different table definitions within the same database.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>For more information about bound databases, refer to Bound Database versus Integrity Enforced.</strong></td>
<td>Interactions Between Btrieve and Relational Constraints (back to Figure 23)</td>
</tr>
</tbody>
</table>
Using Pervasive PSQL Control Center

Table 9  Create New Database GUI Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Related Material</th>
</tr>
</thead>
</table>
| Create dictionary files (if they do not exist)      | Specifies whether you want data dictionary files (DDF) created with the database. Dictionary files are required for relational (SQL) access to your data.  
By default, the dictionary files and data files are created in the same location. You may specify different locations for these types of files after you create the database.  
Typically, the only situation for which you would choose not to create DDF files is when you have an unnamed legacy database for which DDFs already exist, and you are now creating a database name for that database. Under these circumstances, the database engine links the new database name with the pre-existing DDFs. | Dictionary Location (back to Figure 23)                                                                                                    |
| Relational integrity enforced                       | Specifies whether integrity constraints (security, RI, and triggers) are enforced on the database. These constraints apply to Btrieve access to the data files as well as ODBC/SQL access. | Setting Up Referential Integrity Interactions Between Btrieve and Relational Constraints (back to Figure 23) |
| Long Metadata (V2 metadata)                         | The database engine supports two versions of metadata, referred to as version 1 (V1) and version 2 (V2). (V2 metadata is also referred to as “long metadata.”)  
Metadata version is a property of the database and applies to all tables within that database. A database cannot use some tables with V1 metadata and others with V2 metadata. The DDFs of different metadata versions cannot interact. | Versions of Metadata in SQL Engine Reference  
System Tables in SQL Engine Reference (back to Figure 23)                                                                                 |
### Table 9  Create New Database GUI Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Related Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Code Page</td>
<td>Specifies the code page property for the database. The default is “server default,” meaning the default operating system code page on the server.</td>
<td>Database Code Page (back to Figure 23)</td>
</tr>
<tr>
<td></td>
<td>See “Related Material” column for the section where this property is discussed fully.</td>
<td></td>
</tr>
</tbody>
</table>
| Create 32-bit Engine DSN | For ODBC access, you must set up a data source name (DSN) to refer to the database name. By default, the name of the new DSN is the same as the Database Name. Multiple DSNs may point to the same named database. The default is to create the DSN with an encoding translation option of “None.”  
**Note:** The DSN that you are creating is a 32-bit Engine DSN. PCC cannot create 64-bit DSNs. Use an ODBC Interface utility to create 64-bit DSNs (such as ODBC Administrator on Windows). Note that DSN names must be unique for their bitness. A 32-bit DSN and a 64-bit DSN can have the same name because their bitness differs. | DSNs and ODBC Administrator in SQL Engine Reference  
Automatic in SQL Engine Reference  
None in SQL Engine Reference  
Additional Utilities (back to Figure 23) |
Pervasive PSQL Database Tasks

The following tasks pertain to databases.

- To create a new database
- To modify properties of a database
- To delete a database

For conceptual information on named databases, see Pervasive PSQL Database Concepts in Advanced Operations Guide.

To create a new database

Note On Linux, the owner of the directory where you want to create the database must be psql. If not, error message 7039: Dictionary path is invalid results. Use the chown command to change owner of the directory. For example, chown psql directoryname.

1 In PCC Pervasive PSQL Explorer, right-click on the database engine for which you want the new database.

2 Click New ➔ Database.

The Create New Database dialog box appears (see Figure 23).

3 Provide a name for the database and a location (see Table 1, Identifier Restrictions by Identifier Type).

   The name cannot be the same as an existing DSN.

   Also, no two files can share the same file name and differ only in their file name extension if both files are in the same directory. For example, do not name a data file Invoice.btr and another one Invoice.mkd in the same directory. This restriction applies because the database engine uses the file name for various areas of functionality while ignoring the file name extension. Since only the file name is used to differentiate files, files that differ only in their file name extension look identical to the database engine.

4 Specify the additional options on the dialog as required for the database. See Table 9 for a discussion of the options.
To modify properties of a database

1. In PCC Pervasive PSQL Explorer, right-click the database engine for which you want to modify the properties, then click Properties.

2. On the Properties dialog, click the tree node for which you want to specify properties:
   - Directories
   - General
   - Relational Constraints
   - Security

3. Set the specific properties as required.

To delete a database

You cannot delete the database to which you are currently logged in. See To log out from a database engine in Pervasive PSQL User's Guide.

If database security is set to “mixed” or “database,” you must first remove the security. You cannot delete a database if security on it is set to “mixed” or “database.” See To turn off security using Pervasive PSQL Explorer and To turn off security using SQL.

1. In PCC Pervasive PSQL Explorer, right-click on the database engine that you want to delete.

2. Click Delete.
3 Click Yes to confirm the deletion.

If the setting **Always remove associated DSN entries** is turned off, the **Confirm DSN Removal** dialog appears.

![Confirm DSN Removal dialog]

4 If you want to remove a DSN associated with the database, ensure that the DSN is check marked in the list.

Note that multiple DSNs can be associated with one database. In such cases, you may remove the DSNs that you want and retain the ones that you want.

5 Optionally, select the option **Always remove associated DSN entries** if you want PCC to set DSN removal as the default action.

---

**Note** If you select the **Always remove associated DSN entries** option, PCC no longer prompts you for DSN removal when you delete a database. PCC will automatically remove all DSNs associated with the database being deleted.

If you decide later to turn off the option, in PCC click **Window**, expand the **Pervasive** node if it is not already expanded, then click **General**.
Tables

Tables are the objects in which databases store data. Pervasive PSQL contains two types of tables: data and system. Data tables are user-created. Newly created ones are empty and must be populated with data. System tables are created and populated as required by the Pervasive PSQL database management system.

Data Tables

Refer to the chapter Table Editor for a detailed discussion of data tables. That chapter also contains the tasks pertaining to tables, such as creating one, deleting one, working with columns, foreign keys, and so forth.

Note To create a table in a database, database security must be turned off or you must have access rights to create tables.

System Tables

System tables appear in the Pervasive PSQL Explorer under the System Objects node. You may view properties of them as explained in To view properties of a table.

Table Properties

Table properties provides information about the table. Separate tabs let you view general properties, columns information, and indexes information. The following table describes the parameters listed on the General tab.

Table 10 Table Properties on General Tab

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Name</td>
<td>Shows the name of the table as it appears in the database definition.</td>
</tr>
<tr>
<td>Table Location</td>
<td>Shows the physical location of the data file associated with the table.</td>
</tr>
<tr>
<td>Dictionary Path</td>
<td>Shows where the database's DDF files are located.</td>
</tr>
<tr>
<td>File Version</td>
<td>Shows the file format version of the data file.</td>
</tr>
<tr>
<td>Record Length</td>
<td>Shows the length of the data file's records.</td>
</tr>
<tr>
<td>Page Size</td>
<td>Shows the page size (in bytes) of the data file. The page size determines</td>
</tr>
<tr>
<td></td>
<td>the maximum number of index segments that can be defined in a table.</td>
</tr>
<tr>
<td>Number or Records</td>
<td>Shows the number of records currently contained in the data file.</td>
</tr>
</tbody>
</table>
### Table 10  Table Properties on General Tab  

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Indexes</td>
<td>Shows the number of indexes defined for the table.</td>
</tr>
<tr>
<td>Number of Duplicate Pointers (Ptrs)</td>
<td>Shows the number of linked duplicate indexes that can be added.</td>
</tr>
<tr>
<td>Number of Unused Pages</td>
<td>Shows the number of pre-allocated pages available. If pre-allocation is enabled, the MicroKernel pre-allocates a specified number of pages when it creates the data file. Pre-allocation guarantees that disk space for the data file is available when the MicroKernel needs it.</td>
</tr>
<tr>
<td>Variable Records</td>
<td>Shows whether the data file contains variable-length records.</td>
</tr>
<tr>
<td>Variable Record Blank Truncation</td>
<td>Shows whether blank truncation is enabled. If it is, the MicroKernel truncates the blanks in variable-length records. Blank truncation is applicable only if the Variable Records statistic is Yes and Data Compression is set to No.</td>
</tr>
<tr>
<td>Record Compression</td>
<td>Shows whether record compression is enabled. If it is, the MicroKernel compresses each record it inserts into the data file. See Record and Page Compression in Advanced Operations Guide.</td>
</tr>
<tr>
<td>Key Only File</td>
<td>Shows the name of the key-only file for the table, if any. A key-only file contains no data records but serves as an index to other Btrieve files.</td>
</tr>
<tr>
<td>Index Balancing</td>
<td>Shows whether balanced indexing is enabled.</td>
</tr>
<tr>
<td>FreeSpace Threshold</td>
<td>Shows a percentage (5%, 10%, 20% or 30%) if the data file has a free space threshold. The database engine stores the variable-length portions of records on their own pages (called variable pages), separate from the fixed-length portions (which are stored on data pages). The database engine uses the threshold to determine whether to add data to an existing variable page or to create a new one. A higher free space threshold reduces fragmentation of variable-length records across several pages but uses more disk space.</td>
</tr>
<tr>
<td>Uses Alternate Collating Sequence</td>
<td>Shows whether the table uses an alternate collating sequence for sorting.</td>
</tr>
<tr>
<td>System Data Key</td>
<td>Shows whether the data file has system data keys enabled.</td>
</tr>
<tr>
<td>Page Compression</td>
<td>Shows whether page compression is enabled. See Record and Page Compression in Advanced Operations Guide.</td>
</tr>
</tbody>
</table>
Tables

➢ To view properties of a table
1  In Pervasive PSQL Explorer, expand the Tables node.
2  Right-click on the desired table, and click Properties.

Tip You can use the table properties to view a list of the indexed columns for the table.
Data

The tables that you create with PCC are initially empty. You can add data to them through PCC or by importing data. PCC provides a wizard to export data and one to import data.

Creating Data Through PCC

See To add rows of data to the Grid.

Importing Data with Bulk Data Utility

See bdu in Pervasive PSQL User’s Guide.

Importing Data with Import Data Wizard

The Import Data Wizard reads delimited data from a text file and adds the data to a table. The wizard allows you to specify the following:

- Text file that contains the data to import.
- Field delimiter.
- Encoding of the imported data. The encoding must match the encoding that was used to export the data. See Exporting Data with Export Data Wizard.
- Whether or not the first line of the exported data contains the column names. If the data was exported with column names as the first line, it must be imported the same way.

Restrictions

The data must use a field delimiter of the comma, colon, or tab character. A combination of carriage return and line feed must delimit records.

To import data from a database table

1. Right-click a table name under the Tables node.
   
   This is the table into which you want to import the data.
   
2. Click Import Data.
   
3. Provide the import characteristics as discussed above, then click Finish.
**Exporting Data with Export Data Wizard**

The Export Data Wizard exports data from a table to a text file. A combination of carriage return and line feed delimits records.

The wizard lets you specify the following:

- **Name of the file to which the data is exported.** If you include a path with the file name, the directory or directories in the path must already exist.

- **SQL statement on which the export is based.** For example, `SELECT * FROM t1` would export all records from table `t1`.

- **Field delimiter (character used to separate data items in each record).**

- **Encoding of the exported data.** For example, if you select ISO-8859-1, the data is exported using that code page. The encoding choices are obtained from the machine on which the utility is running.

- **Whether or not to write the names of the columns as the first line of the exported data.**

To export data from a database table:

1. Right-click a table name under the **Tables** node.
2. Click **Export Data**.
3. Provide the export characteristics as discussed above then, click **Finish**.
Metadata

Metadata is data about data. The metadata for a Pervasive PSQL database is called a schema. For a relational database, the schema defines the tables, the fields in each table, and the relationships between fields and tables. Schemas are stored as data dictionary files (DDFs) by Pervasive PSQL.

You can export the schema for one or more tables to a text file. The exported schema contains the CREATE TABLE SQL statement (and CREATE INDEX statement if applicable) to create the table and its indexes. The exported file has a default file extension of “sql” and is called an SQL script.

The SQL script file can be run (executed) in SQL Editor. See To open an SQL script.

The SQL script contains the text “Unable to open table” if a table cannot be opened. For example, the error occurs if an owner name is set on a table or a table has been deleted outside of PCC while PCC is currently running.

You have three options when exporting a table schema:

- Include an IN DICTIONARY clause with a USING clause in the statement
- Include only the USING clause in the statement
- Omit both the IN DICTIONARY clause and the USING clause in the statement (a “plain” statement)

IN DICTIONARY Clause

The IN DICTIONARY clause instructs the database engine to modify only the DDFs, which leaves the underlying physical data unchanged. Normally, Pervasive PSQL keeps DDFs and data files synchronized, but this clause allows you to force table dictionary definitions to match an existing data file.

The clause can be useful when you want to create a definition in the dictionary to match an existing data file. Another common use is if you want to duplicate an existing database. You export all the statements from one database, create a new database and then run the exported script against the new database.
If the SQL script contains `IN DICTIONARY` clauses, note that the data file must already exist when you run the SQL script in SQL Editor.

The `IN DICTIONARY` clause is always paired with a `USING` clause. An exported statement looks similar to the following:

```sql
CREATE TABLE "Course" IN DICTIONARY USING 'Course.mkd' (
   "Name" CHAR(7) NOT NULL  CASE ,
   "Description" CHAR(50) CASE ,
   "Credit_Hours" USMALLINT,
   "Dept_Name" CHAR(20) NOT NULL  CASE
);
CREATE UNIQUE INDEX "Course_Name" IN DICTIONARY ON "Course"("Name");
CREATE INDEX "DeptName" IN DICTIONARY ON "Course"("Dept_Name");

See also `IN DICTIONARY` in SQL Engine Reference.
```

**USING Clause**

The `USING` keyword allows you to associate a table with a particular data file.

An exported statement looks similar to the following:

```sql
CREATE TABLE "Course" USING 'Course.mkd' (
   "Name" CHAR(7) NOT NULL  CASE ,
   "Description" CHAR(50) CASE ,
   "Credit_Hours" USMALLINT,
   "Dept_Name" CHAR(20) NOT NULL  CASE
);
CREATE UNIQUE INDEX "Course_Name" IN DICTIONARY ON "Course"("Name");
CREATE INDEX "DeptName" IN DICTIONARY ON "Course"("Dept_Name");

See `USING` in SQL Engine Reference.
```

**Plain Statement**

The “plain” statement omits the `IN DICTIONARY` clause and the `USING` clause. The plain CREATE TABLE syntax is useful to duplicate an existing table by simply changing the table name or to create the same table in a different database.

An exported statement looks similar to the following:
Using Pervasive PSQL Control Center

CREATE TABLE "Course"
(  
  "Name" CHAR(7) NOT NULL  CASE ,  
  "Description" CHAR(50) CASE ,  
  "Credit_Hours" USMALLINT,  
  "Dept_Name" CHAR(20) NOT NULL  CASE 
);  
CREATE UNIQUE INDEX "Course_Name" ON "Course"("Name");  
CREATE INDEX "DeptName" ON "Course"("Dept_Name");  

See CREATE TABLE in SQL Engine Reference.

Exporting a Schema

You can export a schema for a particular table (or tables) or, at the database level, for all tables at once.

➢ To export a table schema for a particular table
1 In Pervasive PSQL Explorer, expand the Tables node for the desired database.
2 Click the table name for which you want to export its schema.
   If you want to export the schema for additional tables, press and hold Shift or Ctrl then click the desired table names. (In other words, use multiple select for the tables you want.)
3 Right-click on the selected table name, then click Export Table Schema.
4 Type the name (and location if you want) for the exported file.
   The default file extension is "sql."
5 Optionally, select the IN DICTIONARY or the USING option.
6 Click OK.
To export all table schemas at once
1 In Pervasive PSQL Explorer, expand the Databases node.
2 Right-click the database name for which you want to export all of the table schemas.
3 Click Export Table Schema.
4 Type the name (and location if you want) for the exported file. The default file extension is “sql.”
5 Optionally, select the IN DICTIONARY or the USING option.
6 Click OK.
Triggers, Stored Procedures, User-defined Functions, and Views

PCC provides a way to create, modify, and delete triggers, stored procedures, user-defined functions, and views. See SQL Editor Tasks.

These objects appear as nodes in the Pervasive PSQL Explorer.

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triggers</td>
<td>A type of stored procedure that are automatically executed when data in a table is modified with an INSERT, UPDATE, or DELETE.</td>
<td>CREATE TRIGGER in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>Stored procedures</td>
<td>A collection of one or more SQL statements that can take and return user-supplied parameters.</td>
<td>CREATE PROCEDURE in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>User-defined functions</td>
<td>A scalar routine that returns a value.</td>
<td>CREATE FUNCTION in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>Views</td>
<td>A database object that stores a query and behaves like a table.</td>
<td>CREATE VIEW in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
</tbody>
</table>
Groups, Users, and Security

Security is a database property that requires a user to provide a user name and password to access the database. By default, database security is turned off.

Database security can be turned on through PCC or by executing an SQL statement. Once enabled, you may create groups and users and assign permissions to them. Permissions can include database rights, table rights, and column rights within tables.

When you turn security on or off, the Master user must have only one connection open and must be the only user connected.

As soon as you turn security on for the first time, only the Master user can access the database. The Master user password, as with all Pervasive PSQL passwords, is case sensitive.

⚠️ **Caution** If you turn on security, be sure to specify a password with a significant length. Do not leave the password field blank because doing so creates a major security risk for your database.

Using Pervasive PSQL Control Center

**Security Tasks** This section contains step-by-step tasks pertaining to security. The tasks are divided into the following categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Tasks</td>
<td>Orient you to the overall use of security</td>
</tr>
<tr>
<td>• To log into a database using PCC when you are already logged into that database as another user</td>
<td></td>
</tr>
<tr>
<td>• To turn on security using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To turn on security using SQL</td>
<td></td>
</tr>
<tr>
<td>• To turn off security using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To turn off security using SQL</td>
<td></td>
</tr>
<tr>
<td>Btrieve Security Policy Tasks</td>
<td>Apply to security policies for the transactional interface</td>
</tr>
<tr>
<td>• To set or change the security policy for a database</td>
<td></td>
</tr>
<tr>
<td>• To use an existing database, including the pre-defined DefaultDB, with your Pervasive PSQL files</td>
<td></td>
</tr>
<tr>
<td>User and Group Tasks</td>
<td>Apply to creating users and groups</td>
</tr>
<tr>
<td>• To create a new group using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To create a new user using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To assign a user to a group using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To delete a group or user using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To work with groups and users using SQL</td>
<td></td>
</tr>
<tr>
<td>Assigning Permissions Tasks</td>
<td>Apply to assigning permissions to users and groups</td>
</tr>
<tr>
<td>• To assign permissions for a group using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To assign permissions for a user using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To assign permissions to all users using Pervasive PSQL Explorer</td>
<td></td>
</tr>
<tr>
<td>• To assign permissions for a group or user using SQL</td>
<td></td>
</tr>
<tr>
<td>Encryption Tasks</td>
<td>Apply to data encryption</td>
</tr>
<tr>
<td>See Data Encryption in Advanced Operations Guide.</td>
<td></td>
</tr>
</tbody>
</table>

**General Tasks**

➤ **To log into a database using PCC when you are already logged into that database as another user**

**Note** As the Master user, logging in as another user can aid you in testing the more restrictive permissions you have assigned this user.
Groups, Users, and Security

1. Right-click on the database name in the PCC Pervasive PSQL Explorer then click **Logout** (name).
   
   Name reflects the name of the user currently logged in to the database. If the database does not have security enabled, name is **Master**. Name may also be **Master** if the current user is logged in as **Master**.
   
   Any nodes expanded for the database are collapsed.

2. Right-click the database name.

3. Click **Login**.

4. Type the user name and password, then click **OK**.

---

**To turn on security using Pervasive PSQL Explorer**

If the database resides on a remote machine, you must provide a user name and password of an administrator or of a member of the Pervasive_Admin group for the remote machine. The user name and password is not required if the database resides on the local machine to which you are logged in (and the local machine is not running Terminal Services).

Turning on security prevents all users from accessing the database unless they login to it using a valid database user name and password. User names and passwords cannot be set up until security is turned on, so the database will be inaccessible to each user for the period of time until you have set up a user account for that user.

1. In Pervasive PSQL Explorer, expand the **Engines** node, then the **Databases** node.

2. Right-click on the desired database then click **Properties**.

3. Click **Security** in the **Properties** tree.

4. Click the **Security** tab.

5. Click **Enable Security** to check mark the option.

6. Type the password you want for **Master Password**, then re-type it for **Confirm Password**.

7. Click **OK**.

   Database security is now on and you are logged in as the **Master** user. For instructions on creating database user accounts, see **User and Group Tasks**.
To turn on security using SQL

You must be logged into the computer as an administrator or as a member of the Pervasive_Admin operating system security group.

Turning on security prevents all users from accessing the database unless they login to it using a valid database user name and password. User names and passwords cannot be set up until security is turned on, so the database will be inaccessible to each user for the period of time until you have set up a user account for that user.

1. Turn security on for the database as explained in General Tasks.

2. In the File menu of PCC, click New » SQL Document (or click in the toolbar).

   The Select Database dialog box appears.

3. Click the database in the list for which you want to create a group or user.

4. Click OK.

5. In SQL Editor, issue the SQL statement SET SECURITY = 'password' where password is the text string you want to use as the password for the Master user.

6. Click SQL » Execute in Text (or click in the toolbar).

   See also SET SECURITY in SQL Engine Reference.
To turn off security using Pervasive PSQL Explorer
You must be logged into the computer as an administrator or as a member of the Pervasive_Admin operating system security group.

Caution Turning off security allow all operating system users to access the database through the relational and transactional interfaces if database security is Mixed or Database mode.

Database user names, passwords, and permissions are retained but not used if security is turn off. If security is re-enabled, the previous user names, passwords, and permissions take effect again. (An exception is the Master user. The Master password is not retained nor re-applied.)

1 In Pervasive PSQL Explorer, expand the Engines node, then the Databases node.
2 Right-click on the desired database then click Properties.
3 Click Security in the Properties tree.
4 Click the Security tab.
5 Click Enable Security to clear the option.
6 Click OK.
   Database security is now off.

To turn off security using SQL

Caution Turning off security allow all operating system users to access the database through the relational and transactional interfaces if database security is Mixed or Database mode.

Database user names, passwords, and permissions are retained but not used if security is turn off. If security is re-enabled, the previous user names, passwords, and permissions take effect again. (An exception is the Master user. The Master password is not retained nor re-applied.)

1 Turn security on for the database as explained in General Tasks.
Using Pervasive PSQL Control Center

2 In the PCC File menu, click **New**  >  **SQL Document** (or click ![image](image.png) in the toolbar).

   The **Select Database** dialog appears.

3 Click the database in the list for which you want to create a group or user.

4 Click **OK**.

5 In SQL Editor, issue the SQL statement `SET SECURITY = NULL`.

6 Click **SQL**  >  **Execute in Text** (or click ![image](image.png) in the toolbar).

   See also `SET SECURITY` in SQL Engine Reference.

**Btrieve Security Policy Tasks**

➤ **To set or change the security policy for a database**

   __Caution__ Changing security policy for a database may prevent current users from accessing the database, if security is turned on and the given users do not have equivalent user accounts and rights under the new security policy.

1 Turn security on for the database as explained in General Tasks.

2 In Pervasive PSQL Explorer, expand the **Engines** node, then the **Databases** node.

3 Right-click the desired database then click **Properties**.

4 Click **Security** in the **Properties** tree.

5 Click the **Btrieve Security** tab.
Groups, Users, and Security

6 Click the desired policy: *Classic*, *Mixed*, or *Database*.

7 Click **OK**.

See also the chapter *Pervasive PSQL Security* in Advanced Operations Guide.

---

**Caution** If your database has security turned on and you change from Classic security policy to Mixed or Database, all users are prevented from accessing the database until you create database user accounts and privileges for them.

---

➢ *To use an existing database, including the pre-defined DefaultDB, with your Pervasive PSQL files*

1 In Pervasive PSQL Explorer, expand the **Engines** node, then the **Databases** node.

2 Right-click on the desired database then click **Properties**.

3 Click **Directories** then click **New**.

4 Type a path for the Pervasive PSQL files then click **OK**.

   If your files are spread over many directories, specify a high-level directory that they all have in common. You can specify a root level if necessary, but doing so includes in the database all Pervasive PSQL+ files at the root level and its subordinate directories.

   You do not need to enter every directory, just the lowest level directory that is common to all Btrieve files you want to include in the database.

5 Turn security on for the database as explained in *General Tasks*.

6 Set permissions for groups and users and explained in *User and Group Tasks*. 
Using Pervasive PSQL Control Center

User and Group Tasks

➤ To create a new group using Pervasive PSQL Explorer
Note that you cannot add a group to another group.
1. Turn security on for the database as explained in General Tasks.
2. Expand the nodes for the database.
3. Right-click the Groups node then click New ▶ Group.
4. Type the name that you want for the group.
5. Click Finish.

➤ To create a new user using Pervasive PSQL Explorer
1. Turn security on for the database as explained in General Tasks.
2. Expand the nodes for the database.
3. Right-click on the Users node then click New ▶ User.
4. Type the name that you want for the user.
5. Type a password for Password and re-type it for Confirm Password.
   Passwords are case sensitive. For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.
6. Optionally, assign the user to a group.
   Click ▼ for Group, then click the desired group in the list.
7. Click Finish.
Groups, Users, and Security

➢ To assign a user to a group using Pervasive PSQL Explorer

Note that a given user cannot be a member of more than one group. All users in a group have exactly the permissions defined for that group. You cannot grant or revoke individual permissions for a user who is a member of a group.

1. Turn security on for the database as explained in General Tasks.
2. If the desired group does not exist, create the group as explained in To create a new group using Pervasive PSQL Explorer.
3. Right-click on a user name under the Users node then click Properties.
4. Click General in the Properties tree.
5. Click for Group, then click the desired group in the list.
6. Click OK.

➢ To delete a group or user using Pervasive PSQL Explorer

Note that a group can be deleted only if no users are assigned to it.

1. Expand the nodes for the database.
2. Expand the Groups node or Users node.
3. Right-click the desired group or user name.
4. Click Delete.
5. Click Yes.

➢ To work with groups and users using SQL

1. Turn security on for the database as explained in General Tasks.
2. In the File menu of PCC, click New SQL Document (or click in the toolbar).
   The Select Database dialog box appears.
3. Click the database in the list for which you want to create a group or user.
4. Click OK.
5. In SQL Editor, create the desired statement for the group or user.
Using Pervasive PSQL Control Center

Refer to the following statements in SQL Engine Reference:

- CREATE GROUP
- ALTER GROUP
- DROP GROUP
- CREATE USER
- ALTER USER
- DROP USER
- GRANT
- REVOKE
- SET PASSWORD

To execute the statement, click SQL ▶ Execute in Text (or click in the toolbar).

Assigning Permissions Tasks

➤ To assign permissions for a group using Pervasive PSQL Explorer

Note Permissions on the Database tab override permissions on the Table tab.

1 Expand the nodes for the desired database.
2 Right-click the group name under the Groups node then click Properties.
3 Click Permissions in the Properties tree.
Click the tab to access permissions for the desired object: database, tables (and columns), stored procedures, or views. See also Permissions on Views and Stored Procedures in SQL Engine Reference.

On the tab, click the option for the desired permission. A check mark indicates that the permission applies.

Click OK.

➢ To assign permissions for a user using Pervasive PSQL Explorer

Note You cannot assign specific permissions to a user if the user is a member of a group. The permissions of the group apply to the user.

Permissions on the Database tab override permissions on the Table tab.

Expand the nodes for the desired database.

Right-click on the user name under the Users node then click Properties.

Click Permissions in the Properties tree.

Click the tab to access permissions for the desired object: database, tables (and columns), stored procedures, or views. See also Permissions on Views and Stored Procedures in SQL Engine Reference.

On the tab, click the option for the desired permission. A check mark indicates that the permission applies.

Click OK.
To assign permissions to all users using Pervasive PSQL Explorer

1. Expand the nodes for the desired database.
2. Right-click on the group **PUBLIC** under the **Groups** node then click **Properties**.
3. Click **Permissions** in the Properties tree.
4. Click the tab to access permissions for the desired object: database, tables (and columns), stored procedures, or views. See also Permissions on Views and Stored Procedures in SQL Engine Reference.
5. On the tab, click the option for the desired permission. A check mark indicates that the permission applies.
6. Click **OK**.

To assign permissions for a group or user using SQL

1. In the **PCC File** menu, click **New > SQL Document** (or click **New SQL Document** in the toolbar). The **Select Database** dialog box appears.
2. Expand the nodes for the desired database.
3. Click **OK**.
4. In SQL Editor, create the desired statement for the group or user. In SQL Engine Reference, see the following:
   - **GRANT**
   - **REVOKE**
   - **SET PASSWORD**
5. Click **SQL > Execute in Text** (or click **SQL Execute in Text** in the toolbar).
Configuration

Configuration is the process by which you provide settings for database engines and clients. You can specify configuration settings with PCC for database engines and the local client.

In PCC, the configuration settings are properties of the engine or client. In Advanced Operations Guide, see To access configuration settings in PCC for an engine and To access configuration settings in PCC for a local client.

In addition, refer to the following sections in Advanced Operations Guide for a discussion of configuration settings that can be configured through PCC:

- Services Configuration Parameters
- Server Configuration Parameters
- Windows Client Configuration Parameters
License Administration

License Models and Working with Keys

This chapter covers the following topics:

- License Administration Concepts
- License Administrator Graphical User Interface
- License Administrator Command Line Interface
- License Administration Tasks
License Administration

License Administration Concepts

This section discusses the following topics:

- License Models
- License Enforcement
- Message Logging

License Models

The Pervasive PSQL products use different license models depending on the product. Pervasive PSQL Server and Workgroup use a user count license model. Pervasive PSQL Vx Server uses a capacity-based license model.

User Count License Model

The user count license model works well for traditional client/server applications in which many users or devices constantly add, update, and delete records from distinct individual desktops. Each product key specifies a licensed user count. A user count allows the specified number of concurrent connections to the Pervasive PSQL database engine. Users are counted by network address. The IP address is used for TCP/IP; the IPX address is used for SPX/IPX.

Each computer that accesses Pervasive PSQL as a client session counts as one user. Multiple applications on a single client computer are counted as one user, not separate users. Internally, Pervasive PSQL assigns each machine a serial number and all connections with the same serial number are recognized as coming from the same machine. A machine with multiple NICs, for example, is recognized as the same machine. Each Terminal Server session also counts as one user.

Collectively, all applications that access the database engine, use the same network protocol and address, and run on the same machine as the database engine count as one user.

Pervasive PSQL Server or Workgroup uses one user count for each unique incoming protocol from the same client computer session. If one application uses TCP/IP and another application uses SPX/IPX, two users are counted if both applications run on the same machine. If different address formats of the same protocol are used, only one user is counted. For example, if one application uses IPv4 and another uses IPv6, only one user is counted if both applications run...
on the same machine. IPv4 and IPv6 are just different address formats of TCP/IP.

**Obtaining a User Count**

An initial user count is provided as part of the product key. A product key is issued by Pervasive Software or by your application vendor if the Pervasive PSQL database engine is embedded in an application.

You may also increase the user count beyond the initial amount provided by the product key. See [Increasing User Count](#).

**Capacity-based License Model**

Pervasive PSQL Vx Server is designed to support highly virtualized environments. For instance, Vx Server includes support for Cloud computing (private, community, and hybrid), as well as for full virtualization, partial virtualization, and paravirtualization. Pervasive PSQL Vx Server has no restrictions on hosting, connection pooling, Internet or intranet use. No additional license is required for use with hypervisor features such as live migration, failover, fault tolerance (FT), high availability (HA), and disaster recovery.

The technological and operational differences that come with virtualized environments require a license model optimized for such environments. The user count license model used for Pervasive PSQL Server and Workgroup does not suffice. That model works well for traditional client/server applications in which many users or devices constantly add, update, and delete records from distinct individual desktops.

Instead, Pervasive PSQL Vx Server uses a license model that shifts the emphasis from how many users to how much work the database server processes. The model is based on capacity to accommodate license enforcement in virtualized environments. For example, each instance of Pervasive PSQL Vx Server has capacity limits on both the number of sessions in use and the data in use. (Also, each instance is identified by a virtual hostname and all virtual MAC addresses.)
Note The information in this section about the capacity-based license model is abbreviated to give you a general understanding of the model. For complete details about the model, see License Model in Pervasive PSQL Vx Product Guide.

Definitions

A “session” is defined as a client ID used by the transactional engine interface or a connection to the relational engine interface. “Client ID” is defined as a 16-byte structure that combines elements provided by the application, by the client platform, and by the database engine to uniquely identify a database transaction context.

The number of sessions in use is a count of the concurrent sessions. For brevity, “number of sessions in use” is also referred to as “session count.” “Session count limit” is the maximum permitted number of concurrent sessions as granted by a license agreement.

“Data in use” is defined as the total size of all concurrently open data files. (A data file is a file created by an application to provide the data processed by that application.) “Data in use limit” is the maximum permitted total size of all concurrently open data files as granted by a license agreement.

Obtaining an Initial Session Count Limit and Data In Use Limit

An initial session count limit and data in use limit is provided as part of the product key. A product key is issued by Pervasive Software or by your application vendor if the Pervasive PSQL database engine is embedded in an application.

You may also increase session count or data in use beyond the initial amount provided by the product key. See Increasing Session Count Limit and Data In Use Limit.

Licensing and Use of Terminal Server

A Terminal Server session is the equivalent of a stand-alone machine. The same criteria used for user count, session count, and data in use for a stand-alone machine also applies to a Terminal Server session.

License Enforcement

Licensing for Pervasive PSQL is enforced by the use of keys. License Administrator is the utility you use to manage those keys. The utility
License Administration Concepts

allows you to authorize and deauthorize keys and to view license information. The utility includes a graphical user interface (GUI) and a command line interface (CLI).

You authorize a key for one of two purposes:

- **Authorize a Product**
- **Increase User Count, Session Count, or Data In Use**

**Authorize a Product**

Product authorization is a key validation process that associates certain machine hardware and machine configurations to the license for a product (called the “product key”). This association results in a unique installation identification (ID) that ensures the copy of software is legitimate and on the appropriate hardware and software platform.

When you authorize a product key, the unique installation ID is sent to Pervasive to verify the authenticity of the key and to ensure that the key is not being used for multiple installations. This process authorizes the product for legitimate use. When the product key is authorized or deauthorized, your remaining authorizations display.

Note that after you have authorized a product on a machine, changes to certain hardware configuration items could disable the key. (Configuration changes on a physical machine such as hard drive serial number, network interface card, MAC address, BIOS firmware, CPU type, operating system running on the hardware, and machine name, and hostname or MAC addresses on a VM.) If you need to change hardware configuration, deauthorize the key first. Doing so disassociates the product key from its unique installation ID. After you complete the hardware configuration changes, you can again authorize the product key.

See also **Key Platforms** for additional details.

**Authorization Access Through A Proxy Server**

If you are using a proxy server, you need to configure it to allow license authorization. Configure the proxy server before you install Pervasive PSQL, or omit product authorization during installation and authorize the product after configuring the proxy server.
Windows

Pervasive Software recommends that you configure proxy servers through Windows Internet Explorer for the best results. Pervasive PSQL license authorization works seamlessly with no modifications needed when proxy servers are configured through Windows Internet Explorer.

If you are currently using a proxy server that was not configured using Windows Internet Explorer, we recommend that you reconfigure it through Windows Internet Explorer. However, if using Windows Internet Explorer to configure your proxy server is not an option, use the following steps.

If you are using a 64-bit Windows operating system, there are two possible places where the Registry settings for proxy servers can be stored:

- HKEY_LOCAL_MACHINE\Software\Pervasive Software\ELS\key
- HKEY_LOCAL_MACHINE\Software\Wow6432Node\Pervasive Software\ELS\key

Depending on the PSQL edition (Server, Client, Vx Server, or Workgroup), the ELS proxy server settings can be created in either location. Licensing manager looks for its proxy server settings first in HKEY_LOCAL_MACHINE\Software\Wow6432Node\Pervasive Software\ELS\key. If it cannot locate them there, it looks in HKEY_LOCAL_MACHINE\Software\Pervasive Software\ELS\key.

Therefore, if you are editing Registry settings for proxy servers, you first need to find the correct location.

1. Begin by looking for a Wow6432Node. If it exists, edit the proxy server Registry settings in the Pervasive Software\ELS\key.
2. If it doesn't, move to the Software directory and edit the proxy server Registry settings in the Pervasive Software\ELS\key.

After locating the correct key, use the following procedure.

1. Modify the Windows Registry and add the following to the Pervasive Software\ELS\key at the correct node:

   String Value          proxy_host
   DWORD Value           proxy_port
License Administration Concepts

Caution Editing the Registry is an advanced procedure. If done improperly, the editing can cause your operating system not to boot. If necessary, obtain the services of a qualified technician to perform the editing. Pervasive Software does not accept responsibility for a damaged Registry.

2 Authorize Pervasive PSQL using License Administrator. See To Authorize a Key for the GUI and To Authorize a Key for the CLI.

**Linux**

1 Manually edit the ELS.INI file to include the following:

   proxy_host
   proxy_port
   proxy_userid
   proxy_passwd

   Note: The proxy_host and proxy_port values are the only ones required by Pervasive PSQL. Depending on the configuration of your proxy server, you may also need to add proxy_userid and proxy_passwd.

2 Authorize Pervasive PSQL Vx Server using the command line interface License Administrator utility. See To Authorize a Key.

**Considerations For No Internet Access**

If the machine on which you are installing Pervasive PSQL does not have Internet access or is not connected to a machine with Internet access, you may authorize the product key using offline authorization.
License Administration

Offline authorization consists of three phases. In the first phase, an Authorization Request Data file (*.ath) is generated on the machine without internet connectivity. During the second phase, the file is then uploaded to an authorization server from a machine with internet access. Upon completion of this transmission, an Authorization Key Data file is downloaded in the form of a product key (*.pky) file. In the final phase, the product key file is authorized on the machine without Internet connectivity and Pervasive PSQL is authorized.

The quickest and simplest method for authorizing Pervasive PSQL is using online or remote authorization. Whenever possible, try to authorize your Pervasive PSQL product using one of these methods. The Pervasive PSQL installation, for example, allows you to authorize online as part of the installation process.

**Note** Offline and telephone authorization are not available for Pervasive PSQL Vx Server.

**Key Status, or State**

The status of a product key is displayed in the License Administrator as the “state.” Key states include the following:

- **Active** – key is authorized and available for use.
- **Inactive** – key is no longer in use. This state is reserved for situations where the key is incompatible with the current engine. Examples would include previous versions of Pervasive PSQL, such as v10 and prior on v11, or a Workgroup key on a Server engine, or 32 bit key on a 64-bit engine.
- **Expired** – key is no longer available for use. Temporary keys move to the Expired state once their evaluation period has passed.
- **Disabled** – key is no longer available for use. If a key is disabled because of a failed validation, it can be restored to active by fixing the problem and then performing a validation on the key.
- **Failed Validation** – key is usable but only for a set number of days. See Failed Validation and Failed-Validation Period.
License Administration Concepts

Failed Validation and Failed-Validation Period

If a key fails validation, you have a set number of days to remedy the condition(s) that caused it to fail. This period of time, sometimes referred to as the “failed-validation period,” is reported as a date in the Expiration Date column of License Administrator. See GUI Visual Reference. If the condition for the failed validation is not corrected by the date, the key changes state to disabled.

After you implement changes to correct the failed validation, perform a Validation action. See To Perform a Validation. The validation, among other actions, verifies the current machine signature and key combination. If the conditions were corrected, the state of the key changes to active.

More than one condition at a time can cause a failed validation. You can correct what you think is the sole condition yet the key still remains in the failed validation state. If so, refer to the PVSW.LOG file for more information. That log file contains all of the conditions causing the failed validation. See Pervasive PSQL Event Log.

If the changes that caused the failed validation are due to intentional hardware upgrades performed without prior deauthorization of the key, the appropriate course of action is a repair of the key.

Repair

License Administrator provides the functionality to perform a repair on a key that is disabled or has failed validation. Moreover, it not only repairs the key, it automatically deauthorizes the key and then authorizes it again on the same machine.

You can perform a repair using either the graphical user interface (see License Administrator Graphical User Interface) or the command line interface (see License Administrator Command Line Interface).

If the problem is more significant than a change in hardware, for example, the machine on which a key is installed no longer boots, you cannot repair the key. In that case, contact support for assistance.

Increase User Count, Session Count, or Data In Use

Situations can arise in which you want to increase user count, session count, or data in use from the initial amount provided by the
permanent product key. You increase the amount by authorizing an increase key. Multiple increase keys can be authorized on a product key.

The increase is immediately available when you authorize the key. A restart of the database engine is not required.

To authorize an increase key, you must already have a permanent product key present on the system. The software vendor ID number for an increase key must match the vendor ID number for an installed permanent key. Increase keys obtained from Pervasive Software are universal. That is, they are compatible with any permanent key from any vendor.

Also note the restrictions for key platforms listed in Tables 11 and 12.

**Increasing User Count**

You increase the user count by authorizing a user count increase (UCI) key. See To Authorize a Key and To Determine a Total User Count. UCI keys are provided by Pervasive Software or an application vendor.

If you deauthorize a permanent product key, all user count increase keys associated with that permanent key are also deauthorized. If you deauthorize a UCI key, only the user count increase associated with that UCI key is deauthorized.

Trial versions of the software include a trial license that have a set user count that cannot be increased.

**Increasing Session Count Limit and Data In Use Limit**

Pervasive Software or an application vendor offers a key to increase session count limit, data in use limit, or both. See To Authorize a Key, To Determine the Session Count Limit and To Determine the Data In Use Limit.

If you deauthorize the permanent product key, all increase keys associated with that permanent key are also deauthorized. If you deauthorize an increase key, only the session count increase or data in use increase associated with that key is deauthorized.

Trial versions of the software include a trial license that have a set value for session count and data in use that cannot be increased.
Key Platforms

The product key controls the conditions under which the product permits installation of, or access to, the database engine. The following tables summarize the restrictions based on platform for Windows and Linux. For example, if your key has a platform of "Win64," you can authorize that key on a database engine running only on a Windows 64-bit platform. The key is invalid for Windows 32-bit platforms and for any Linux platform.

Note that versions of Pervasive PSQL Server and Pervasive PSQL Vx Server are available for each bit architecture on each operating system. Pervasive PSQL Workgroup is a 32-bit product that runs only on Windows.

Table 11 Restrictions Pertaining to Windows Platforms

<table>
<thead>
<tr>
<th>Key Platform</th>
<th>Pervasive PSQL Server</th>
<th>Pervasive PSQL Vx Server</th>
<th>OS Bit Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Windows32</td>
<td>Windows64</td>
<td>32-bit</td>
</tr>
<tr>
<td>Any</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Windows</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Win32</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Win64</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

¹The Windows 32-bit version of Pervasive PSQL Server or Pervasive PSQL Vx Server can be installed on a Windows 64-bit machine but must run under the Windows-on-Window (WOW) execution layer.

Table 12 Restrictions Pertaining to Linux Platforms

<table>
<thead>
<tr>
<th>Key Platform</th>
<th>Pervasive PSQL Server</th>
<th>Pervasive PSQL Vx Server</th>
<th>OS Bit Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linux32</td>
<td>Linux64</td>
<td>32-bit</td>
</tr>
<tr>
<td>Any</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Linux</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Linux32</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

¹The Windows 32-bit version of Pervasive PSQL Server or Pervasive PSQL Vx Server can be installed on a Windows 64-bit machine but must run under the Windows-on-Window (WOW) execution layer.
License Administration

Table 12  Restrictions Pertaining to Linux Platforms

<table>
<thead>
<tr>
<th>Key Platform</th>
<th>Pervasive PSQL Server</th>
<th>Pervasive PSQL Vx Server</th>
<th>OS Bit Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux32</td>
<td>Linux64</td>
<td>32-bit</td>
<td></td>
</tr>
<tr>
<td>Linux64</td>
<td>✓</td>
<td>32-bit</td>
<td>✓</td>
</tr>
</tbody>
</table>

1The Linux 32-bit version of Pervasive PSQL Server or Pervasive PSQL Vx Server can be installed and run on a Linux 64-bit machine.

Message Logging

Messages pertaining to licensing are logged to the various logging repositories used by Pervasive PSQL, such as Notification Viewer. See Pervasive PSQL Message Logging.
License Administrator Graphical User Interface

The graphical user interface (GUI) License Administrator runs only on Windows platforms and allows you to authorize and deauthorize keys, increase user counts, session count limit, data in use limit, and view license information. You can start the GUI as a stand-alone application or from within Pervasive PSQL Control Center (PCC). You can administer licenses for local or remote database engines with the GUI.

See also GUI Tasks for the tasks specific to the graphical user interface.

GUI Visual Reference

The following image shows the GUI. The table below the image describes the GUI objects. Click on an area of the image for which you want more information.
## License Administration

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Name</td>
<td>By default, shows the name of the server that the License Administrator is accessing. This object is also an entry field into which you can type the name of a server. If you have typed in a server name but not clicked <strong>Connect</strong> (or pressed <strong>Enter</strong>), the field can contain a server name that differs from the one in the title bar. The title bar always shows the server that License Administrator is accessing. The <strong>Server Name</strong> field can also be populated if you click <strong>Browse</strong> and choose a server.</td>
<td>To Select a Server for License Administration (Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Connect</td>
<td>Establishes communication between License Administrator and the computer listed in the <strong>Server Name</strong> field.</td>
<td>To Select a Server for License Administration (Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Browse</td>
<td>Allows you to choose a server on which to administer licenses.</td>
<td>To Select a Server for License Administration (Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Key</td>
<td>Accepts a key (a combination of letters and numbers) typed in or pasted.</td>
<td>License Administration Concepts To Authorize a Key (Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Authorize</td>
<td>Authorizes the key entered in the <strong>Key</strong> field onto the specified database engine. The Server Name field shows the server on which the database engine is running. This button is enabled if the key in the <strong>Key</strong> field is valid and a database engine is running on the specified computer.</td>
<td>Authorize a Product Increase User Count, Session Count, or Data In Use To Authorize a Key (Return to GUI Dialog image)</td>
</tr>
</tbody>
</table>
License Administrator Graphical User Interface

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>License Information</td>
<td>Lists the license information for the specified computer:</td>
<td>License Models</td>
</tr>
<tr>
<td></td>
<td>• Product – The name of the Pervasive PSQL product, such as Server, Vx Server, or Workgroup.</td>
<td>To Display License Information</td>
</tr>
<tr>
<td></td>
<td>• Product Key – The Product Key number associated with the corresponding product license.</td>
<td>(Return to GUI Dialog image)</td>
</tr>
<tr>
<td></td>
<td>• State – The status of the Product Key. Values are Active, Inactive, Expired, Disabled and Failed Validation. (A license can be disabled if certain hardware configuration items have changed since the key was applied.)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Platform – The combination of Pervasive PSQL product, operating system, and bit architecture to which the license applies. See Tables 11 and 12.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• License Type – The type of license, such as permanent, temporary or an increase license for user count, session count, or data in use.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• User Count – The user count for the specific license.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Session Count – The session count limit for the specific license.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Data In Use GB – The data in use limit in gigabytes for the specific license.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Expiration Date – The date on which a license expires, if applicable, or “n/a” if not applicable.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Vendor – A unique number identifying the vendor software that installed the license.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Application – A unique number identifying the application to which the license applies.</td>
<td></td>
</tr>
<tr>
<td>Hide Expired</td>
<td>Hides all expired licenses in the list of licenses for the current session of License Administrator. This checkbox is selected by default.</td>
<td>To Hide Expired Licenses</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Deauthorize</td>
<td>Deauthorizes the selected key from the specified server. If the selected key is a permanent key, also deauthorizes all increase keys (user count, session count, data in use) associated with the permanent key. This button is enabled when a Product name is selected in the License Information list.</td>
<td>To Deauthorize a Key</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Repair</td>
<td>Repairs, deauthorizes, and authorizes the selected key on the specified server.</td>
<td>Repair</td>
</tr>
<tr>
<td></td>
<td>This button is enabled when a Product name is selected in the License Information list.</td>
<td>To Repair a Key</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Return to GUI Dialog image)</td>
</tr>
</tbody>
</table>
License Administration

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Authorizations</td>
<td>Displays the remaining number of authorizations for all product keys and the</td>
<td>To Display Remaining Authorizations</td>
</tr>
<tr>
<td></td>
<td>remaining number of repairs that a user can perform on a key in a state of</td>
<td>(Return to GUI Dialog image)</td>
</tr>
<tr>
<td></td>
<td>Failed Authorization or Disabled. This information also displays automatically</td>
<td></td>
</tr>
<tr>
<td></td>
<td>at the time of authorization and deauthorization, if applicable.</td>
<td></td>
</tr>
<tr>
<td>Help</td>
<td>Displays the online help for License Administrator.</td>
<td>To Display Help</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Return to GUI Dialog image)</td>
</tr>
<tr>
<td>Close</td>
<td>Closes the License Administrator Utility</td>
<td>(Return to GUI Dialog image)</td>
</tr>
</tbody>
</table>
License Administrator Command Line Interface

The command line interface (CLI) runs on all platforms supported by Pervasive PSQL. The CLI is functionally equivalent to the GUI. You can administer licenses for local or remote database engines with the CLI.

The GUI and the CLI list information about the authorized licenses, such as the type of license, user count, session count limit, data in use limit, platform, expiration date, and so forth. License key information is also available from the installation media.

**CLI Syntax**

The CLI uses the same syntax on all platforms:

```
utility_name -option parameters
```

The utility name is one of the following:

- `clilcadm.exe` (Windows 32-bit)
- `w64clilcadm.exe` (Windows 64-bit)
- `clilcadm` (Linux 32-bit)
- `clilcadm64` (Linux 64-bit)
The following table describes the options and parameters.

Table 13  License Administrator Command Line Options and Parameters

<table>
<thead>
<tr>
<th>Options and Parameters</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>--&lt;interpret</td>
<td>i&gt; [key]</td>
</tr>
<tr>
<td></td>
<td>• Product – The name of the Pervasive PSQL product, such as Server, Vx Server, or Workgroup.</td>
</tr>
<tr>
<td></td>
<td>• Product Key – The Product Key associated with the corresponding product license.</td>
</tr>
<tr>
<td></td>
<td>• State – The status of the Product Key. Values are Active, Inactive, Expired, Disabled and Failed Validation. (A license may be disabled if certain hardware configuration items have changed since the key was applied.)</td>
</tr>
<tr>
<td></td>
<td>• Platform – The combination of Pervasive PSQL product, operating system, and bit architecture to which the license applies. See Tables 11 and 12.</td>
</tr>
<tr>
<td></td>
<td>• License Type – The type of license, such as permanent, temporary or an increase license for user count, session count, or data in use.</td>
</tr>
<tr>
<td></td>
<td>• User Count – The user count for the specific license. See also User Count License Model.</td>
</tr>
<tr>
<td></td>
<td>• Session Count – The session count limit for the specific license. See also Capacity-based License Model.</td>
</tr>
<tr>
<td></td>
<td>• Data In Use GB – The data in use limit in gigabytes for the specific license. See also Capacity-based License Model.</td>
</tr>
<tr>
<td></td>
<td>• Expiration Date – The date on which a license expires, if applicable, or &quot;n/a&quot; if not applicable.</td>
</tr>
<tr>
<td></td>
<td>• Vendor – A unique number identifying the vendor software that installed the license.</td>
</tr>
<tr>
<td></td>
<td>• Application – A unique number identifying the application to which the license applies.</td>
</tr>
<tr>
<td>--&lt;authorize</td>
<td>a&gt; &lt;key</td>
</tr>
<tr>
<td>--&lt;deauthorize</td>
<td>d &gt; key</td>
</tr>
</tbody>
</table>
License Administrator Command Line Interface

Table 13  License Administrator Command Line Options and Parameters

<table>
<thead>
<tr>
<th>Options and Parameters</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>`&lt;clear</td>
<td>c&gt; [key] [force]`</td>
</tr>
<tr>
<td>`&lt;generate</td>
<td>g&gt; key output_filename`</td>
</tr>
<tr>
<td>`&lt;number</td>
<td>n&gt; [key]`</td>
</tr>
<tr>
<td>`&lt;servername</td>
<td>s&gt; &lt;servername&gt;`</td>
</tr>
<tr>
<td>`&lt;validate</td>
<td>t&gt;`</td>
</tr>
<tr>
<td>`&lt;username</td>
<td>u&gt;`</td>
</tr>
<tr>
<td>`&lt;password</td>
<td>p&gt; &lt;password&gt;`</td>
</tr>
<tr>
<td>`&lt;help</td>
<td>h&gt;`</td>
</tr>
</tbody>
</table>
License Administration

Table 13  License Administrator Command Line Options and Parameters continued

<table>
<thead>
<tr>
<th>Options and Parameters</th>
<th>Function</th>
</tr>
</thead>
</table>
| -<repair | e> key | Repairs and authorizes a key (if repair is possible). The Repair action requires the following conditions to be met:  
  • The key to be repaired must be compatible with the version of the database engine currently installed.  
  • The key to be repaired must be in a “disabled” or a “failed validation” state.  
  • A permanent key cannot be in the “active” state.  
  • There must be internet access to communicate with the Pervasive license server.  
  • The allowable number of repairs must not have been exceeded.  
  For a successful repair, the message “key has been repaired” is returned. |
| no option or wrong option | Displays information about the options and parameters. |

See CLI Tasks for the tasks pertaining to the CLI.
License Administration Tasks

This section explains the tasks that you can perform for license administration. The majority of tasks involve License Administrator. Some tasks, such as telephone authorization, are accomplished by other means.

License Administrator GUI Tasks

- To Start License Administrator from Pervasive PSQL Control Center (PCC)
- To Start License Administrator as a Stand-alone Application
- To Select a Server for License Administration
- To Authorize a Key
- To Display Remaining Authorizations
- To Hide Expired Licenses
- To Deauthorize a Key
- To Repair a Key
- To Display License Information
- To Determine a Total User Count
- To Refresh the License Information List
- To Display Help

CLI Tasks

- To Display Information About a Key
- To Perform a Validation
- To Display Remaining Authorizations
- To Authorize a Key
- To Deauthorize a Key
- To Remove Keys Without Deauthorizing
- To Repair a Key
- To Display Help

Alternative Authorization Tasks

- To Authorize a Key Remotely
- To Deauthorize a Key Remotely
License Administration

- To Authorize a Key Offline
- To Deauthorize a Key Offline
- To Authorize a Key Using the Pervasive Phone Authorization Wizard
- To Authorize a Key Using the Command-line Phone Authorization Utility

Notification Viewer

- To View Licensing Messages in Logging Repositories

GUI Tasks

➢ To Start License Administrator from Pervasive PSQL Control Center (PCC)

1. In PCC, click **Tools** then **License Administrator**.

   License Administrator accesses the selected server to determine the type of license required.

   If you are not logged in to the server, a dialog appears to log in to the operating system (not the database engine).

   Login with a user name that has administrative privileges on the operating system where the database engine is running.

   Once logged in, the server name appears in the License Administrator title bar and in the **Server Name** field.

Note

For servers running a previous version of Pervasive PSQL, use the license utility provided with that version of the product.

If the server is not the one you want, select a different server as described in **To Select a Server for License Administration**.

➢ To Start License Administrator as a Stand-alone Application

1. Access **License Administrator** from the operating system **Start** menu or **Apps** screen.
License Administration Tasks

This step assumes that License Administrator was installed as part of a Pervasive PSQL default installation. A vendor application that uses an embedded database engine may require that you start License Administrator with different menu commands. (The GUI executable is named guilcadm.exe.)

The utility connects to the local server on which the database engine is running.

If the server is not the one you want, select a different server as described in To Select a Server for License Administration.

To Select a Server for License Administration

The database engine must be running on the computer you specify so that License Administrator can connect to the engine.

1. In the Server Name field, type the name of the server for which you want to administer database licenses.

As an alternative to typing a server name, you may click Browse to display a list of network servers from which to select. Click on the server for which you want to administer database licenses, then click OK.

2. Click Connect (or press Enter).

License Administrator accesses the specified server to determine the type of license required.

- If you are not logged in to the server, a dialog appears to log in to the operating system (not the database engine).

  On the login dialog, type the name of an operating system user for User Name. The user must have administrative privileges for the database engine. For Password, type the appropriate password for the user name you specified. Click Login. The server name appears in the License Administrator title bar and in the Server Name field.

Note For servers running a previous version of Pervasive PSQL, use the license utility provided with that version of the product.
To Authorize a Key

Note that the GUI License Administrator can be used for online authorization of a local machine connected to the Internet, and for remote authorization of the database engine on a machine not connected to the Internet.

1 Ensure that the server name in the title bar is the server for which you want to authorize a database engine key. If not, select a different server as described in To Select a Server for License Administration.

2 Type, or paste, the key into the Key field.
   Lowercase letters are automatically converted to uppercase.

3 Press Enter or click the button to authorize the key.
   The Key field is cleared if the key is successfully authorized.

4 Verify the new license information that appears in the License Information list. (If necessary, press F5 to refresh the list.)
   The license is now active. You do not have to restart the database engine.

Note When authorizing an increase key for user count, session count, or data in use, a permanent license must already exist. The vendor ID of the increase key must match the vendor ID of the permanent key.

Tip Product authorization is tied to your machine’s hardware configuration. After you have authorized a product key on a machine, changes to certain hardware configuration items could disable the key. If you need to change hardware configuration, deauthorize the key first. Deauthorizing the key disassociates the product key from the unique hardware configuration. After you complete the hardware configuration changes, you can again authorize the product key.
License Administration Tasks

➤ **To Display Remaining Authorizations**

1. Ensure that the server name in the title bar is the correct server for this operation. If not, select a different server as described in To Select a Server for License Administration.

2. Click **Remaining Authorizations**.
   
The number of remaining authorizations displays for all product keys.

➤ **To Hide Expired Licenses**

1. Ensure that the server name in the title bar is the server for which you want to hide expired licenses. If not, select a different server as described in To Select a Server for License Administration.

2. Click **Hide Expired**, if it is not already selected.
   
The list of licenses changes to exclude all expired licenses.

   _______________________________________________________________________
   **Note** The **Hide Expired** checkbox is selected by default and can be selected or cleared as needed.
   _______________________________________________________________________

➤ **To Deauthorize a Key**

   _______________________________________________________________________
   **Note** Temporary licenses cannot be deauthorized. They expire at the end of their evaluation period.
   _______________________________________________________________________

1. Ensure that the server name in the title bar is the server from which you want to deauthorize a database license. If not, select a different server as described in To Select a Server for License Administration.

2. Click a **Product** name in the **License Information** list.
   
   You can select multiple keys by holding down the **Shift** or **Control** keys and clicking the desired product names.

3. Click the button to deauthorize the key.
Note You cannot deauthorize a key in a failed validation state. Either rectify the issue before deauthorizing or call Pervasive Software support for assistance.

Tip When you deauthorize a product key, all increase keys for user count, session count, or data in use associated with that product key are automatically deauthorized as well.

To Repair a Key

Tip After you have applied a product key on a machine, changes to the hardware configuration can cause a key to fail validation or become disabled, requiring a repair. To avoid the need for a repair, deauthorize the key before changing the hardware configuration and authorize the key again after the new configuration is complete.

1. Ensure that the server name in the title bar is the server for which you want to repair a key. If not, select a different server as described in To Select a Server for License Administration.

2. In the License Information list, click the Product name of the product with the key you want to repair.

3. Click Repair.

4. In the dialog box that opens, confirm that this is a genuine copy of the software product. When the repair is done, a completion message is displayed.

To Display License Information

1. Ensure that the server name in the title bar is the server for which you want to administer database licenses. If not, select a different server as described in To Select a Server for License Administration.

2. View the information for the authorized licenses in the License Information list.
License Administration Tasks

Note You must first authorize a key to display information about the associated license, such as user count, session count, data in use, license type, expiration date and so forth. Once authorized, if the license is not what you want, you may deauthorize it provided the license it is not a temporary license. Temporary licenses expire on their own and cannot be deauthorized.

See To Authorize a Key and To Deauthorize a Key.

➢ To Determine a Total User Count

1. Ensure that the server name in the title bar is the server for which you want determine a database user count. If not, select a different server as described in To Select a Server for License Administration.

2. For a particular product, total the user count values that appear in the License Information list in the “User Count” column. Exclude any values for expired temporary licenses.

   For example, total all values for the Pervasive PSQL Server product for the permanent license, all user count increases, and any temporary licenses that have not expired.

   The sum is the total number of users who can concurrently connect to the database engine on the server.

Note An easier way to determine the total user count is with Monitor. See Viewing Resource Usage in Advanced Operations Guide. The “Maximum” value for User Count is the total user count for the permanent license, all user count increase licenses, and any temporary licenses that have not expired.

➢ To Determine the Session Count Limit

1. Ensure that the server name in the title bar is the server for which you want determine a session count limit. If not, select a different server as described in To Select a Server for License Administration.
For a particular product, total the session count values that appear in the License Information list in the “Session Count” column. Exclude any values for expired temporary licenses.

For example, total all values for Pervasive PSQL Vx Server for the permanent license, all session count increases, and any temporary licenses that have not expired. The sum is the total number of sessions that can concurrently connect to the database engine on the server for that particular product (the session count limit).

Note An easier way to determine the session count limit is with Monitor. See Viewing Resource Usage in Advanced Operations Guide. The “Maximum” value for Session Count is the session count limit for the permanent license, all session count increase licenses, and any temporary licenses that have not expired.

To Determine the Data In Use Limit

1 Ensure that the server name in the title bar is the server for which you want to determine a data in use limit. If not, select a different server as described in To Select a Server for License Administration.

2 For a particular product, total the data in use values that appear in the License Information list in the “Data In Use GB” column. Exclude any values for expired temporary licenses.

For example, total all values for Pervasive PSQL Vx Server for the permanent license, all data in use increases, and any temporary licenses that have not expired. The sum is the maximum permitted amount in gigabytes for all concurrently open data files for that particular product.

Note An easier way to determine the data in use limit is with Monitor. See Viewing Resource Usage in Advanced Operations Guide. The “Maximum” value for Data In Use MB is the data in use limit for the permanent license, all data in use increase licenses, and any temporary licenses that have not expired.

Be aware that License Administrator uses gigabytes (GB) as the
units because that is how data in use is associated with a key. Monitor uses megabytes (MB) as the units for resource usage and kilobytes (KB) for the size of an individual file. The different contexts require units appropriate for each context.

To Refresh the License Information List

1. Ensure that the server name in the title bar is the server you want. If not, select a different server as described in To Select a Server for License Administration.

2. Press the F5 key.

The information for applied licenses is re-displayed, and any information is cleared from the Key field.

The Server Name field is cleared then displays the machine name to which License Administrator is currently connected. For example, suppose License Administrator is connected to PVSW1 and you attempt to connect to PVSW2, which does not have a database engine running. License Administrator displays a message that it could not connect, and PVSW2 is displayed in the Server Name field. After you click the F5 key, PVSW1 is displayed in the field.

To Display Help

1. Click Help from the menu bar of the License Administrator. The License Administrator Graphical User Interface section opens. It enables you to identify and understand the purpose of the objects on the GUI.

2. To consult other areas of documentation about the License Administrator, access the Table of Contents for the documentation and click the desired area:
   - License Administration Concepts — to understand the basics of License Administrator
   - License Administrator Command Line Interface — to understand the options and parameters for the CLI
   - License Administration Tasks — to perform activities with License Administrator.
CLI Tasks

Note For all of the following tasks, the Windows 64-bit version is named w64clicadm, and the Linux 64-bit version is named clicadm64.

➢ To Display Information About a Key
1 To display information about all keys, enter the following command:
   clicadm -i
2 To display information about a specific key, enter the following command:
   clicadm -i key

See License Administrator Command Line Options and Parameters for the columnar information returned by the -i option.

➢ To Perform a Validation
1 To perform a validation, enter the following command:
   clicadm -t

Tip To determine the product key’s post-validation state and date, use the -i option to display information about the specified key.

➢ To Display Remaining Authorizations
1 To display the remaining authorizations for all product keys, enter the following command:
   clicadm -n
2 To display the remaining authorizations for a specific key, enter the following command:
   clicadm -n key
License Administration Tasks

**Tip** To determine a product key, use the -i option to display information about all keys. See To Display Information About a Key.

---

**To Authorize a Key**

1. Enter the following command:
   ```
   cliLcadm -a key
   ```

**Tip** When you authorize an increase key for user count, session count, or data in use, a permanent product key must already exist. The vendor ID of the increase key must match the vendor ID of the permanent key.

---

**To Deauthorize a Key**

1. Enter the following command:
   ```
   cliLcadm -d key
   ```

**Tip** When you deauthorize a product key, all increase keys for user count, session count, or data in use associated with that product key are automatically deauthorized as well.

---

**To Remove Keys Without Deauthorizing**

Circumstances can occur for which you need to remove a key and its associated increase keys for user count, session count, or data in use without deauthorizing the product. This action is called “clearing” a key.

1. To clear a specific key, enter the following command, where key is the individual key you want to clear:
   ```
   cliLcadm -c key
   ```

2. To clear all keys, enter the following command:
   ```
   cliLcadm -c
   ```
License Administration

With either command, you can include the “force” parameter if you want to suppress the clear confirmation message.
Example: clilcdm -c force

➢ To Repair a Key
1 Enter the following command:
   clilcdm -e key
2 The command window displays a message asking you to confirm that this is a genuine copy of the software product. Enter “A” to confirm. After the repair is complete, the command window displays a message confirming the repair.

➢ To Display Help
1 Enter either of the following commands:
   clilcdm -<help | h>
   clilcdm ?

Alternative Authorization Tasks
Pervasive PSQL provides some alternate ways to authorize and deauthorize your Pervasive PSQL product. If possible, however, authorize Pervasive PSQL using online authorization. If the machine where Pervasive PSQL is installed has only remote internet access, you can authorize remotely.

Tip See To Authorize a Key for information on online authorization.

➢ To Authorize a Key Remotely
Note that the GUI License Administrator can also be used for remote authorization if you connect to a remote database engine.
1 At a command prompt, enter the following command:
   clilcdm -a key -<server | s> servername -<username | u>
   username -<password | p> password
If the machine where Pervasive PSQL is installed does not have internet access, offline authorization is available.
License Administration Tasks

➢ To Deauthorize a Key Remotely

1. At a command prompt, enter the following command:
   ```
   clilcadm -d key -<server | s> servername -<username | u> username -<password | p> password
   ```

   **Tip** When you deauthorize a product key, all increase keys (user count, session count, or data in use) associated with that product key are automatically deauthorized as well.

➢ To Authorize a Key Offline

Only consider offline authorization if the machine on which you are running Pervasive PSQL Server or Workgroup does not have internet access. Offline authorization is completed in three phases using two machines. Machine A does not have Internet connectivity and machine B does have Internet connectivity. In this example machine A is the machine that needs to authorize the Pervasive PSQL product key.

**Note** Offline authorization is not available for Pervasive PSQL Vx Server.

Phase 1 - Generate Authorization Request Data

1. At machine A (without Internet connectivity), enter the following at a command prompt:
   ```
   clilcadm -<generate | g> product_key output_filename.ath
   ```

2. Save the file to a portable storage device or in a location that can be easily accessed by the machine that has Internet connectivity.

   **Tip** If the machine with Internet connectivity does not have Pervasive PSQL installed, you must also copy the licgetauth.exe file from the Pervasive PSQL bin folder to the portable storage device or remote location.

Phase 2 - Transmit Authorization Request Data and Retrieve Authorization Key Data
License Administration

1 At machine B (with Internet connectivity), connect to the portable storage device or access the location where you saved the Authorization Request Data file.

2 Enter the following at a command prompt, where output_filename.ath is the file you created in step 1 of Phase 1:
   licgetauth.exe output_filename.ath
   The above command retrieves the Authorization Key Data used to authorize the key and the product. The authorization key data is saved as a .pky file to the same location where you saved the authorization request file.

   Tip If you saved the Authorization Request Data and Authorization Key Data files to a hard drive, you will need to copy these files to either a portable storage device or a remote location.

Phase 3 - Install Authorization Key Data

1 Back at machine A (without Internet connectivity), connect once more to the portable storage device or access the location where the authorization key data was saved.

2 Enter the following at a command prompt, where authorization_filename.pky is the file created in step 2 of Phase 2:
   clicadm -a authorization_filename.pky
   Pervasive PSQL is now authorized.

➢ To Deauthorize a Key Offline

Deauthorizing a key offline requires that you contact Pervasive Support.

Telephone Authorization

In the event that it is not possible to authorize keys online, remotely, or offline, you can authorize Pervasive PSQL Server or Workgroup using telephone authorization. Telephone authorization is not available for Pervasive PSQL Vx Server.

Telephone authorization requires that you have completed a successful installation of Pervasive PSQL Server or Workgroup and
To authorize a key via telephone authorization, you can use the command-line utility or the Pervasive Phone Authorization Wizard. The Phone Authorization Wizard is available on Windows and Linux and guides you through the steps necessary to authorize your Pervasive PSQL product.

Tip When authorizing the Workgroup Engine, you need to elevate the privileges of the database engine before authorizing a key with a licensing utility.

➢ To Authorize a Key Using the Pervasive Phone Authorization Wizard

1 Ensure that you have completed a successful installation of Pervasive PSQL Server or Workgroup and have a valid product key. See also Telephone Authorization.

2 At a command prompt on the machine where the engine is installed, type guiPaadm.

➢ To Authorize a Key Using the Command-line Phone Authorization Utility

1 Ensure that you have a valid product key. See To Display Information About a Key. If you do not have a product key, you may obtain one from Pervasive Software.

2 At a command prompt on the machine where the engine is installed, type the utility name appropriate for your platform:
   • clipaadm.exe (Windows 32-bit)
   • w64clipaadm.exe (Windows 64-bit)
   • clipaadm (Linux 32-bit)
   • clipaadm64 (Linux 64-bit)

3 At the prompt, enter the product key you want to authorize.
License Administration

The Pervasive PSQL Phone Authorization Utility returns an authorization code you will use to complete telephone authorization.

4 Perform one of the following actions to complete telephone authorization:

a. If you obtained your product key from Pervasive Software, call 1-800-287-4383 (U. S. toll-free) or 00800.1212.3434 (Europe toll-free) to complete telephone authorization with Pervasive Software personnel. The Pervasive Software staff will provide you with an authorization key data string which you will enter as the final step of authorization.

b. If you did not obtain your product key from Pervasive Software, contact the vendor or original equipment manufacturer (OEM) from whom you obtained the key. The vendor or OEM personnel will assist you to complete telephone authorization.

Telephone Deauthorization

Telephone deauthorization is not available. If you authorized Pervasive PSQL using telephone authorization, you must call Pervasive Software to deauthorize.

Notification Viewer

The Pervasive Notification Viewer is a utility for displaying messages logged by the licensing components. The purpose of the utility is to inform you of noteworthy licensing messages in a noticeable but unobtrusive manner. Although not part of License Administrator, Notification Viewer is mentioned here because of its allied purpose pertaining to licensing.

➢ To View Licensing Messages In Logging Repositories

1 See the following topic for the repository in which you are interested:

   • Pervasive Notification Viewer
   • Operating System Event Log
   • Pervasive PSQL Event Log
A Tour of Pervasive Table Editor

The topics in this chapter include:

- Table Editor Concepts
- Table Editor Graphical User Interface
- Table Editor Tasks
Table Editor

Table Editor Concepts

This section contains the following topics:

- Overview
- Table Editor Pages
- Data Types
- Null Values

Overview

Table Editor is one of the editor windows within Pervasive PSQL Control Center (PCC). Table Editor is a special type of editor that contains multiple pages. The pages are represented by tabs across the bottom of the editor. The editor allows you to add, delete, or change the characteristics of columns within a table. The table may be one newly created or an existing table that you want to edit.

To modify tables with Table Editor, you must have full administrator rights on the machine on which the database engine is running even if you are a member of the Pervasive_Admin group. See Granting Administrative Rights for the Database Engine and Database Security in Advanced Operations Guide.

Caution
Backup all your data definition files (DDFs) and data files before you perform functions through Table Editor. This tool gives you the ability to modify your database table definitions and data. If you inadvertently set the options incorrectly or enter incorrect data, you could change your files in an irreversible manner. Full recovery is possible if you have performed a backup.

Table Editor Pages

Table Editor contains the following pages that you use as work areas:

- Columns
- Indexes
- Foreign Keys
- SQL View

You select a page by clicking on its page name tab.
Table Editor Concepts

Note Save your changes before switching pages.

Columns Page
The Columns page lets you add, delete, modify columns, and set primary keys. See Columns Page for a description of the areas on the Columns page. See Columns Tasks for the tasks that you perform on the page.

Data Types
Refer to Pervasive PSQL Supported Data Types in SQL Engine Reference for a list of the data types supported by the database engine. That section lists the Pervasive PSQL data types for the transactional and relational interfaces and the equivalent ODBC data types. You may use any data types listed in Pervasive PSQL Supported Data Types and that appear in the “Type” selection list on the Columns page in Table Editor.

Null Values
The ability to modify the null attribute of a column is subject to the following restrictions:

- The target column cannot have a PRIMARY/FOREIGN KEY constraint defined on it.
- If converting the old type to the new type causes an overflow (arithmetic or size), the ALTER TABLE operation is aborted.
- If a nullable column contains NULL values, the column cannot be changed to a non-nullable column.

If you must change the data type of a key column, you can do so by deleting the index key, changing the data type, and re-adding the key. Keep in mind that you must ensure that all associated index key columns in the database remain synchronized.

For example, if you have a primary index key in table T1 that is referenced by foreign keys in tables T2 and T3, you must first delete the foreign keys. Then you delete the primary key and change all three columns to the same data type and size. Finally, you must re-add the primary key and then the foreign keys.
For additional information on nulls, see the following:

- **INSERT** in SQL Engine Reference
- **Null Value** in Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

**Indexes Page**

The Indexes page lets you add and modify indexes and index segments. See Indexes Tasks for the tasks that you perform on the page.

**Foreign Keys Page**

The Foreign Keys page lets you add and modify foreign keys. See Foreign Keys Tasks for the tasks that you perform on the page.

**SQL View Page**

The SQL View page lets you view the CREATE TABLE and ALTER TABLE statements that apply to the table. Note that SQL View is display only. You cannot modify the SQL statements but you can copy them. See SQL View Tasks for the tasks that you perform on the page.

The SQL View page reflects changes made on the other pages as explained below.

**CREATE Statements**

The CREATE TABLE statement shows the SQL used to create the table. For a new table, one that has not yet been saved, the CREATE TABLE statement reflects the SQL used to implement edits from the other three tabbed dialogs. For example, if you were to create a new table named “MyNewTable,” SQL View initially shows the following CREATE statement:

```sql
CREATE TABLE MyNewTable (  );
```
If you were to add two CHAR columns to the new table on the Columns page, SQL View reflects this in the CREATE statement:

```sql
CREATE TABLE MyNewTable(
    "FirstName" CHAR(20),
    "LastName" CHAR(30)
);
```

If a table has been saved (already exists), the CREATE TABLE statement shows the SQL required to create the table. For example, if you were to edit the "Course" table provided with the sample database DEMODATA, SQL View shows the following CREATE statements:

```sql
CREATE TABLE Course(
    "Name" CHAR(7) NOT NULL ,
    "Description" CHAR(50),
    "Credit_Hours" USMALLINT,
    "Dept_Name" CHAR(20) NOT NULL
);
CREATE INDEX Course_Name ON Course("Name");
CREATE UNIQUE INDEX DeptName ON Course("Dept_Name");
```

**ALTER Statements**

When you are editing an existing table, the ALTER TABLE statements show what SQL is used to implement edits from the other three Table Editor pages. For example, suppose that you edit the "Course" table provided with the sample database DEMODATA. On the Index page, you change the sort order for the "Name" index segment from ascending to descending. SQL View shows the following ALTER statements:

```sql
DROP INDEX Course.Course_Name;
CREATE INDEX Course_Name ON Course("Name" DESC);
```

Saving a table clears the ALTER TABLE statements because no changes are pending.
Table Editor Graphical User Interface

The Table Editor graphical user interface (GUI) provides work areas (called pages) for the following:

- **Columns Page**
- **Indexes Page**
- **Foreign Keys Page**
- **SQL View Page**

**Columns Page**

The Columns page can be considered the primary page of Table Editor just as columns and rows are central to tables. The following image shows the Columns page of Table Editor. The table below the image describes the GUI objects. Click on an area of the image for which you want more information.

See also **Columns Tasks** for the tasks that you perform on the page.
<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column Name</td>
<td>Specifies the alpha-numeric name for the column.</td>
<td>To specify a column name&lt;br&gt;Relational Interface Limits in SQL Engine Reference&lt;br&gt;Versions of Metadata in SQL Engine Reference&lt;br&gt;Identifier Restrictions by Identifier Type in Advanced Operations Guide (Columns Page image)</td>
</tr>
<tr>
<td>Type</td>
<td>Specifies the data type of the column.</td>
<td>Pervasive PSQL Supported Data Types in SQL Engine Reference&lt;br&gt;To set a column data type (Columns Page image)</td>
</tr>
<tr>
<td>Size</td>
<td>Specifies how many bytes are permitted for the data type. A shaded cell indicates that size does not apply.</td>
<td>To set a column size (Columns Page image)</td>
</tr>
<tr>
<td>Precision</td>
<td>Specifies the number of significant digits for floating point values. A shaded cell indicates that precision does not apply.</td>
<td>To set column precision (Columns Page image)</td>
</tr>
<tr>
<td>Scale</td>
<td>Specifies the number of significant digits that are to the right of the decimal point for floating point values. A shaded cell indicates that scale does not apply.</td>
<td>To set a column scale (Columns Page image)</td>
</tr>
<tr>
<td>Null</td>
<td>Specifies whether NULL values are allowed for the data type. A shaded square (□) indicates that nulls do not apply to the data type. A checked square (☑) indicates that NULL values are allowed for the data type.</td>
<td>To set a column to allow or disallow nulls (Columns Page image)</td>
</tr>
<tr>
<td>Case</td>
<td>Specifies whether the database engine uses case-sensitive or case-insensitive comparisons when searching for character values in the database. A checked square (☑) indicates that case-insensitive values are used. A shaded square (□) indicates that case sensitivity does not apply to the data type.</td>
<td>To set case sensitivity for a column (Columns Page image)</td>
</tr>
</tbody>
</table>
Table Editor

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collate</td>
<td>Specifies that an alternating collating sequence (ACS) is used for sorting. Contains the path and ACS file name.</td>
<td>To set a column collating sequence (Columns Page image)</td>
</tr>
<tr>
<td>Default</td>
<td>Specifies a default value for the column. The default value is used if you perform an SQL INSERT for a row but do not provide a value for the column.</td>
<td>To set a column default (Columns Page image)</td>
</tr>
</tbody>
</table>

**Indexes Page**

The Indexes page allows you to add and delete indexes. The Indexes page is explained within the context of the tasks that you perform for indexes. See Indexes Tasks.

**Foreign Keys Page**

The Foreign Keys page allows you to add and delete foreign keys. The Foreign Keys page is explained within the context of the tasks that you perform for foreign keys. See Foreign Keys Tasks.

**SQL View Page**

The SQL View page displays, and allows you to copy, the SQL statements used to create or alter the table. The CREATE TABLE panel displays the SQL with which you could create the same table. The ALTER TABLE panel reflects any editing changes to an existing table that you make with Table Editor. When you save the table changes, the ALTER TABLE panel is cleared and the ALTER statement(s) becomes part of the CREATE TABLE statement.

The SQL View page is further explained within the context of the tasks that you can perform. See SQL View Tasks.
Table Editor Tasks

This section explains the tasks that you perform with Table Editor. The tasks are divided into the following categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Tasks</td>
<td>Orient you to the overall use of Table Editor</td>
</tr>
<tr>
<td>Columns Tasks</td>
<td>Apply to using the Columns page</td>
</tr>
<tr>
<td>Indexes Tasks</td>
<td>Apply to using the Indexes page</td>
</tr>
<tr>
<td>Foreign Keys Tasks</td>
<td>Apply to using the Statistic page</td>
</tr>
<tr>
<td>SQL View Tasks</td>
<td>Apply to using the SQL View page</td>
</tr>
</tbody>
</table>

*Note* You cannot save the changes to the structure of a table if any queries in SQL Editor are holding the table “open.” Close the SQL Editor holding open the table then save the changes.

**General Tasks**  General tasks apply to the overall use of the tool.

**Getting Started**

- To start Table Editor for an existing table
- To start Table Editor for a new table
- To work with columns
- To work with indexes
- To work with foreign keys
- To view SQL statements applicable to the table
Table Editor

Data
- To view table data
- To identify tables with changes that have not been saved
- To save changes for the table being edited
- To save changes for all tables being edited
- To undo changes or to redo changes

Columns Tasks Column tasks apply to the Columns page.
- To insert a column between existing columns
- To insert a column at the end
- To select a column or multiple columns
- To delete a column
- To specify a column name
- To set a column data type
- To set a column size
- To set column precision
- To set a column scale
- To set a column to allow or disallow nulls
- To set case sensitivity for a column
- To set a column collating sequence
- To set a column default
- To set or remove a column as a primary key
Table Editor Tasks

Indexes Tasks
Index tasks apply to the Indexes page.
- To create an index
- To create a unique index
- To create a partial index
- To modify an existing index
- To delete an index
- To insert an index segment
- To modify an index segment
- To delete an index
- To arrange the order of index segments
- To specify a sort order for an index
- To allow duplicates in an index
- To specify index as modifiable

Foreign Keys Tasks
Foreign keys tasks apply to the Foreign Keys page.
- To add a foreign key
- To modify a foreign key
- To delete a foreign key

SQL View Tasks
SQL view tasks apply to the SQL View page.
- To copy SQL statements
- To maximize or restore view of SQL statements

General Tasks

➤ To start Table Editor for a new table

1 Start PCC if it is not already running. (See Starting PCC on Windows.)

2 Expand the Engines and Databases nodes in Pervasive PSQL Explorer.

3 Right-click on the database to which you want to add the new table.

4 Click New ➤ Table and type the name for new table.
Tip For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.

Note that in the same directory, no two files should share the same file name and differ only in their file name extension. For example, do not create a table (data file) Invoice.btr and another one Invoice.mkd in the same directory. This restriction applies because the database engine uses the file name for various areas of functionality while ignoring the file name extension. Since only the file name is used to differentiate files, files that differ only in their file name extension look identical to the database engine.

5 Click Finish.

➢ To start Table Editor for an existing table
1 Start PCC if it is not already running. (See Starting PCC on Windows.)
2 Expand the Engines and Databases nodes in Pervasive PSQL Explorer.
3 Right-click on the table that you want to modify then click Edit.

➢ To work with columns
1 Perform the steps for To start Table Editor for an existing table or To start Table Editor for a new table.
2 Click the Columns page tab.

➢ To work with indexes
1 Perform the steps for To start Table Editor for an existing table or To start Table Editor for a new table.
2 Click the Indexes page tab.

➢ To work with foreign keys
1 Perform the steps for To start Table Editor for an existing table or To start Table Editor for a new table.
2 Click the Foreign Keys page tab.
Table Editor Tasks

➢ To view SQL statements applicable to the table
1 Perform the steps for To start Table Editor for an existing table or To start Table Editor for a new table.
2 Click the SQL View page tab.

➢ To view table data
1 If the Grid window view is not displayed, click Window ➔ Show View ➔ Grid.
2 Perform the steps for To start Table Editor for an existing table or To start Table Editor for a new table.
3 By default, the Grid shows all of the data for the table (the result of a SELECT * FROM table statement).
   If the Grid is empty but the table contains data, right-click on any row in the Grid then click Refresh.
   Note that the Grid allows you to directly change database data by changing the values in the grid cells. See Grid Tasks.

➢ To identify tables with changes that have not been saved
1 Observe the Table Editor tab at the top. The tab contains the name of the table being created or edited. An asterisk (*) precedes the name if any modifications have occurred to columns, indexes, or foreign keys but not yet saved.
Table Editor

➢ To save changes for the table being edited

Note that you cannot undo or redo changes to a table once the table has been saved.

1. Click File ➔ Save or click ⌘.

Note You cannot save the changes to the structure of a table if the table is open in SQL Editor. Close the SQL Editor referencing the table then save the changes.

➢ To save changes for all tables being edited

Note that you cannot undo or redo changes to tables once the tables have been saved.

1. Click File ➔ Save All.

Note You cannot save the changes to the structure of a table if the table is open in SQL Editor. Close the SQL Editors referencing the tables then save the changes.

➢ To undo changes or to redo changes

1. In the toolbar, click ⌘ to undo an action; click ⌘ to redo an action.

If multiple actions have occurred since the last save, you can repeatedly click the undo or redo toolbar buttons. When no more actions are available for undo or redo, the toolbar button becomes disabled.

Note that you cannot undo or redo changes to a table once the table has been saved.
Columns Tasks

➢ **To insert a column between existing columns**

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for *To work with columns*.

2. Right-click on an existing column row above which you want to insert a new column.

3. Click **Insert Column**.

   The new column appears above the existing column row. The default name of the inserted column is “columnn,” where “n” is a number that automatically increments by one. (The first column you insert is column0, the second column1 and so forth.)

   **Tip** You can also insert a column by clicking on an existing column row, then pressing **Ctrl+Insert** or clicking ![Insert](insert.png). Repeating either action inserts a series of columns in succession.

4. Click **File ➤ Save** or ![Save](save.png) before changing pages within Table Editor.

➢ **To insert a column at the end**

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for *To work with columns*.

2. Right-click anywhere on the empty column row below the last existing column row, then click **Add Column**.

   or

   Click the “Column Name” cell on the empty column row below the last existing column row and start typing a name for the column.

   The default name of the inserted column is “columnn,” where “n” is a number that automatically increments by one. (The first column you insert is column0, the second column1 and so forth.)
Table Editor

Tip You can also insert a column at the end by clicking or pressing Ctrl+Insert.

Repeating either action inserts a series of columns in succession. The insert action automatically adds the new column to the end.

3 Click File ▶ Save or before changing pages within Table Editor.

➢ To select a column or multiple columns
1 Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.
2 Click (the column selection icon on the far left of the column row) for the desired column.

To select multiple columns, press and hold Shift or Ctrl then click for the desired additional columns.

➢ To delete a column
1 Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.
2 Right-click anywhere on the desired column row.
3 Click Drop Column.

Tip You can also delete a column by clicking on an existing column row, then pressing Ctrl+Delete or clicking .

You can also delete multiple columns by selecting multiple columns rows. See To select a column or multiple columns.

4 Click File ▶ Save or before changing pages within Table Editor.

➢ To specify a column name
1 Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.
2 Click in the Column Name cell for the desired column.
3. Delete the existing column name.
4. Type the name you want.

Tip For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.

Also, as a general rule, avoid using reserved words for column names. See Reserved Words in SQL Engine Reference. See also Versions of Metadata in SQL Engine Reference.

5. Click File ➤ Save or before changing pages within Table Editor.

To set a column data type

The data in your database is converted if you change a column data type. For a listing of data types, see Pervasive PSQL Supported Data Types in SQL Engine Reference.

Changing a column data type sets the defaults for that type on the following: size, scale, precision, default, and collate.

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.
2. Click in the Type cell for the desired column.
3. Open the list for data types (click).
4. Scroll to the data type you want. (You may also type the first letter of the desired data type to scroll. Repeatedly typing the first letter scrolls to each data type that begins with that letter.)
5. Click the data type you want.
6. Click File ➤ Save or before changing pages within Table Editor.

Note Changing a data type on a column that contains a default value causes the default value to be cleared; reset the value if necessary.
To set a column size

Data in your database is truncated if you change the column to a smaller size for the following data types:

- CHAR
- NUMERIC
- VARCHAR

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2. Click in the Size cell for the desired column.
   
   You can set a size only for applicable data types, such as CHAR. If size is not applicable, the grid cell is shaded and you will be unable to edit the Size.

3. Delete the existing size.

4. Type the size you want.

5. Click File > Save or before changing pages within Table Editor.
Table Editor Tasks

➢ To set column precision

Precision specifies the number of significant digits for floating point values.

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2. Click in the Precision cell for the desired column.

You can set precision only for applicable data types, such as DECIMAL. If precision is not applicable, the grid cell is shaded and you will be unable to edit the Precision.

3. Delete the existing value.

4. Type the precision value you want.

5. Click File ➤ Save or before changing pages within Table Editor.

➢ To set a column scale

Scale specifies the number of significant digits that are to the right of the decimal point for floating point values.

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2. Click in the Scale cell for the desired column.

You can set a scale value only for applicable data types, such as NUMERIC. If scale is not applicable, the grid cell is shaded and you will be unable to edit the Scale.

3. Delete the existing value.

4. Type the scale value you want.

5. Click File ➤ Save or before changing pages within Table Editor.
To set a column to allow or disallow nulls


1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2. Click the option box in the **Null** cell for the desired column.

   You can allow nulls only for applicable data types. A shaded square indicates that null values do not apply to the data type.

<table>
<thead>
<tr>
<th>Option State</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>Nulls specified</td>
</tr>
<tr>
<td></td>
<td>Nulls allowed but not specified</td>
</tr>
<tr>
<td></td>
<td>Nulls do not apply</td>
</tr>
</tbody>
</table>

Also see Null Values.

3. Click File ➤ Save or before changing pages within Table Editor.

To set case sensitivity for a column

Case sensitivity does not apply if the key uses an alternate collating sequence (ACS). You cannot specify case sensitivity and use an ACS.

1. Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2. Click the option box in the **Case** cell for the desired column.

   You can set a collating sequence only for applicable data types. A shaded square indicates that case sensitivity does not apply to the data type.

<table>
<thead>
<tr>
<th>Option State</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>Case insensitive</td>
</tr>
<tr>
<td></td>
<td>Case sensitive</td>
</tr>
<tr>
<td></td>
<td>Case does not apply</td>
</tr>
</tbody>
</table>
Table Editor Tasks

By default, Pervasive PSQL is case sensitive when sorting string keys. Uppercase letters are sorted before lowercase letters. If you specify case insensitive, values are sorted without distinguishing case.

3 Click File ➤ Save or before changing pages within Table Editor.

➢ To set a column collating sequence

For additional information about collating sequences, see Manipulating Btrieve Data Files with Maintenance in Advanced Operations Guide and Alternate Collating Sequences in the Pervasive PSQL Programmer’s Guide, which is part of the Pervasive PSQL Software Developer’s Kit (SDK).

If you use an alternate collating sequence (ACS), you cannot specify case sensitivity. Case sensitivity does not apply if the key uses an ACS.

1 Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2 Click in the Collate cell for the desired column.

You can set an alternating collating sequence (ACS) only for applicable data types. If collating sequence is not applicable, the grid cell is shaded and you will be unable to edit the Collate cell.

3 Delete the existing value, if present.
4  Type the path and ACS file name you want.

Pervasive PSQL supplies an ACS file, upper.alt, in the Samples folder. (See Where are the Pervasive PSQL files installed? in Getting Started With Pervasive PSQL.) To use this file, you would type `file_path\PSQL\samples\upper.alt`.

Upper.alt treats upper and lower case letters the same for sorting. For example, if a database has values `abc`, `ABC`, `DEF`, and `Def`, inserted in that order, the sorting with upper.alt returns as `abc`, `ABC`, `DEF`, and `Def`. (The values `abc` and `ABC`, and the values `DEF` and `Def` are considered duplicates and are returned in the order in which they were inserted.) Normal ASCII sorting sequences upper case letters before lower case, such that the sorting would return as `ABC`, `DEF`, `Def`, `abc`.

5  Click File ➤ Save or ✒ before changing pages within Table Editor.

➢ To set a column default

The default value is used if you perform an SQL INSERT for a row but do not provide a value for the column.

1  Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2  Click in the Default cell for the desired column.

You can set a default only for applicable data types. If a default is not applicable, the grid cell is shaded and you will be unable to edit the Default.

3  Delete the existing value, if present.

4  Type the default value you want.
Table Editor Tasks

The column default can be a scalar function for certain data types:

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Scalar Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>• now()</td>
</tr>
<tr>
<td></td>
<td>• curdate()</td>
</tr>
<tr>
<td></td>
<td>See also NOW()</td>
</tr>
<tr>
<td></td>
<td>and CURDATE()</td>
</tr>
<tr>
<td></td>
<td>, both in SQL</td>
</tr>
<tr>
<td></td>
<td>Engine Reference.</td>
</tr>
<tr>
<td>Time</td>
<td>• now()</td>
</tr>
<tr>
<td></td>
<td>• curtime()</td>
</tr>
<tr>
<td></td>
<td>See also NOW()</td>
</tr>
<tr>
<td></td>
<td>and CURTIME()</td>
</tr>
<tr>
<td></td>
<td>in SQL Engine Reference.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>• now()</td>
</tr>
<tr>
<td></td>
<td>See also NOW()</td>
</tr>
<tr>
<td></td>
<td>in SQL Engine Reference.</td>
</tr>
</tbody>
</table>

1 The names are case insensitive. NOW() and now() are equivalent. The parentheses are required. That is, NOW is invalid but NOW() is valid.

5 Click File » Save or before changing pages within Table Editor.

➢ To set or remove a column as a primary key

Note that you cannot set a primary key on a column that allows NULLs.

1 Ensure that the “Columns” page of Table Editor is active. If required, perform the steps for To work with columns.

2 Click (the column selection icon on the far left of the column row) for the desired column(s).

To select multiple columns, press and hold Shift or Ctrl then click for the desired additional columns.

3 Click (the primary key icon).

If the column(s) is not a primary key, the action sets the column(s) as a primary key.

If the column, or if any of the columns when multiple columns are selected, is already a primary key, the action removes the setting from all columns.
Table Editor

For example, suppose that column 1 is a primary key and you want columns 1, 2, and 3 to be the primary key. You press and hold Ctrl then click columns 1, 2, and 3. When you click the primary key icon, it is removed from column 1 but not added to columns 2 and 3. If you click the primary key icon again, then columns 1, 2, and 3 are designated as primary keys.

4 Click File > Save or before changing pages within Table Editor.

Index Tasks

➢ To create an index

Only the database engine can add an index to an IDENTITY or SMALLIDENTITY column. However, you can include an IDENTITY or SMALLIDENTITY column as part of a multiple-segment index.

Table Editor permits you to include an IDENTITY or a SMALLIDENTITY column in the Indexes list if you have not saved the table. However, the DBMS returns an error when you attempt to save the table. After you delete the IDENTITY or SMALLIDENTITY column from the list, you may then save the table.

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2 Click Add.

   The New Index dialog displays.

3 Type the name of the new index and click OK.

   Tip For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.

   The new index appears in the Indexes list and the Index Segment Details display. Note that the first column is populated into the Columns list.

   Note New indexes are created by default as Normal.
4 In the **Columns** list, select the Column to designate for the Index or Index Segment.

**Caution** Indexes must have at least one Column designated. If you do not select a Column for the Index, the first column remains selected.

The default sort order is Ascending. If you need a descending sort order, select **Descending** from the Sort Order list for the Column you want changed.

**Note** Some data types, such as LONGVARBINARY, cannot be used for an index. Columns with such data types are not valid choices.

5 Continue selecting columns from the list until all the segments are added.

Note that a column can be selected for an index only once. Once selected, the column is removed from the list of choices because the column has already been designated in the index.

6 Click **File** ➤ **Save** or before changing pages within Table Editor.

See also **Creating Indexes** in Pervasive PSQL Programmer’s Guide, which is part of the Pervasive PSQL Software Developer’s Kit (SDK).
To create a unique index

Only the database engine can add an index to an IDENTITY or SMALLIDENTITY column. However, you can include an IDENTITY or SMALLIDENTITY column as part of a multiple-segment index.

Table Editor permits you to include an IDENTITY or a SMALLIDENTITY column in the Indexes list if you have not saved the table. However, the DBMS returns an error when you attempt to save the table. After you delete the IDENTITY or SMALLIDENTITY column from the list, you may then save the table.

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2 Click Add.
   The New Index dialog displays.

3 Type the name of the new index and click OK.

Tip For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.

The new index appears in the Indexes list and the Index Segment Details display.

Note New indexes are created by default as Normal.
4 Select **Unique** in the Index Segment Details area to designate the index as unique.

Selecting Unique disables duplicatability, restricting duplicates. Note that the first column is populated into the Columns list.

5 In the **Columns** list, select the Column to designate for the Index or Index Segment.

**Caution** Indexes must have at least one Column designated. If you do not select a Column for the Index, the first column remains selected.

The default sort order is Ascending. If you need a descending sort order, select **Descending** from the Sort Order list for the Column you want changed.

**Note** Some data types, such as LONGVARBINARY, cannot be used for an index. Columns with such data types are not valid choices.

6 Continue selecting columns from the list until all the segments are added.

Note that a column can be selected for an index only once. Once selected, the column is removed from the list of choices because the column has already been designated in the index.
Table Editor

7 Click **File > Save** or ![icon] before changing pages within Table Editor.

See also **Creating Indexes** in Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

➢ **To create a partial index**

Only the database engine can add an index to an IDENTITY or SMALLIDENTITY column. However, you can include an IDENTITY or SMALLIDENTITY column as part of a multiple-segment index.

Table Editor permits you to include an IDENTITY or a SMALLIDENTITY column in the Indexes list if you have not saved the table. However, the DBMS returns an error when you attempt to save the table. After you delete the IDENTITY or SMALLIDENTITY column from the list, you may then save the table.

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for **To work with indexes**.

2 Click **Add**.

The **New Index** dialog displays.

3 Type the name of the new index and click **OK**.

---

**Tip** For a list of database object lengths and invalid characters, see **Identifier Restrictions by Identifier Type** in Advanced Operations Guide.

---

The new index appears in the Indexes list and the Index Segment Details display.

---

**Note** New indexes are created by default as Normal.

---

4 Select **Partial** in the Index Segment Details area to designate the index as partial.

Note that the first column is populated into the Columns list.
5 In the **Columns** list, select the Column to designate for the Index or Index Segment.

**Caution** Indexes must have at least one Column designated. If you do not select a Column for the Index, the first column remains selected.

The default sort order is Ascending. If you need a descending sort order, select **Descending** from the Sort Order list for the Column you want changed.

**Note** Partial Indexes are restricted to columns with a data type of CHAR or VARCHAR and that are designated as the only or last segment in an Index.

6 Continue selecting columns from the list until all the segments are added.

Note that a column can be selected for an index only once. Once selected, the column is removed from the list of choices because the column has already been designated in the index.

7 Click **File > Save** or ![Save button](Save) before changing pages within Table Editor.

See also **Creating Indexes** in Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).
To modify an existing index

The database engine creates some indexes, such as IDENTITY column indexes and primary key indexes. These indexes are read-only and cannot be modified.

Table Editor permits you to include an IDENTITY or a SMALLIDENTITY column in the Indexes list if you have not saved the table. However, the DBMS returns an error when you attempt to save the table. After you delete the IDENTITY or SMALLIDENTITY column from the list, you may then save the table.

1. Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.
2. Click the desired index in the Indexes list.
   The Index Segment Details display.
3. Modify the segment details as desired.
4. Click File → Save or before changing pages within Table Editor.

To delete an index

The database engine creates some indexes, such as IDENTITY column indexes and primary key indexes. These indexes are read-only and cannot be deleted.

1. Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.
2. Click the desired index in the Indexes list.
3. With the Index you want to delete selected, click Delete in the Indexes list.
4. Click Yes to confirm the deletion.
5. Click File → Save or before changing pages within Table Editor.
To insert an index segment

For detailed information about segments, see Segmentation in the Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

Only the database engine can add an index to an IDENTITY or a SMALLIDENTITY column. However, you can include an IDENTITY or a SMALLIDENTITY column as part of a multiple-segment index. See also AUTOINC in SQL Engine Reference.

1. Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2. Click the desired index in the Indexes list.

   The Index Segment Details displays and lists the selected Index Segments.

3. Click in the first empty Columns cell, then open the Columns list (click ▼).

4. From the list, click the desired column for the segment.

   Note Some data types, such as LONGV ARBINARY, cannot be used for an index. Columns with such data types are not valid choices.

The default sort order is Ascending. If you need a descending sort order, select Descending from the Sort Order list for the Column you want changed.
Table Editor

5  Continue selecting columns from the list until all the segments are added.
   Note that a column can be selected for an index only once. Once selected, the column is removed from the list of choices because the column has already been designated in the index.

6  Click File → Save or before changing pages within Table Editor.

➢ To modify an index segment

For detailed information about segments, see Segmentation in the Pervasive PSQL Programmer’s Guide, which is part of the Pervasive PSQL Software Developer’s Kit (SDK).

Only the database engine can add an index to an IDENTITY or a SMALLIDENTITY column. However, you can include an IDENTITY or a SMALLIDENTITY column as part of a multiple-segment index. See also AUTOINC in SQL Engine Reference.

1  Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2  Click the desired index in the Indexes list.
   The Index Segment Details displays.

3  Click the desired segment in the Columns cell then open the list of columns (click).

4  From the Columns list, click the desired column designated as the index segment.
Table Editor Tasks

**Note** Some data types, such as LONGVARBINARY, cannot be used for an index. Columns with such data types are not valid choices.

The default sort order is Ascending. If you need a descending sort order, select **Descending** from the Sort Order list for the Column you want changed.

5 Continue selecting columns from the list until you have completed all segment modifications.

Note that a column can be selected for an index only once. Once selected, the column is removed from the list of choices because the column has already been designated in the index.

6 Click **File > Save** or before changing pages within Table Editor.

➢ **To delete an index segment**

**Note** Each index requires a minimum of one segment. To delete an index that has only one segment, delete the index itself.

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for **To work with indexes**.

2 Click the desired index in the Indexes list.

The Index Segment Details displays, listing all the designated index segments.

3 Click the desired index segment.

4 With the index segment you want to delete selected, click **Delete** in the Index Segment Details Columns list.
5 Click **File ➤ Save** or ➤ before changing pages within Table Editor.

➢ **To arrange the order of index segments**

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for **To work with indexes**.

2 Click on the desired index in the Indexes list.

The Index Segment Details displays.

3 Click the index segment you want to reorder.

4 Click **Up** to move the segment toward the top of the segment grouping, or **Down** to move the segment toward the bottom.

5 Click **File ➤ Save** or ➤ before changing pages within Table Editor.
Table Editor Tasks

➢ To specify a sort order for an index

For detailed information about sort order, see Sort Order in the Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2 Click on the desired index in the Indexes list.

   The Index Segment Details displays.

3 Click in the Sort Columns cell then open the list of sort choices (click \(\uparrow\)).

4 From the Sort Order list, click \textit{Ascending} or \textit{Descending}.

   The default sort order when an index segment is created is ascending.

5 Click \textit{File} » \textit{Save} or \(\textit{\textbullet\textbullet}\) before changing pages within Table Editor.

➢ To allow duplicates in an index

For detailed information about duplicates, see Duplicatability in the Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2 Click the desired index in the Indexes list.

   The Index Segment Details displays.

3 Clear the \textit{Unique} option in the Index Segment Details area by selecting one of the other options (\textit{Partial} or \textit{Normal}).

\begin{quote}
\textbf{Note} By default, indexes are created as Normal, allowing duplicates.
\end{quote}

4 Click \textit{File} » \textit{Save} or \(\textit{\textbullet\textbullet}\) before changing pages within Table Editor.
Table Editor

➢ To specify index as modifiable

For detailed information about modifiability, see Modifiability in the Pervasive PSQL Programmer's Guide, which is part of the Pervasive PSQL Software Developer's Kit (SDK).

1 Ensure that the “Indexes” page of Table Editor is active. If required, perform the steps for To work with indexes.

2 Click the desired index in the Indexes list.

   The Index Segment Details displays.

3 Click Allow Modifications option.

<table>
<thead>
<tr>
<th>Index Segment Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow Modifications</td>
</tr>
</tbody>
</table>

   A check mark in the box indicates that the index value can be modified. Lack of a check mark indicates that the index value cannot be modified.

<table>
<thead>
<tr>
<th>Option State</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>Values can be modified</td>
</tr>
<tr>
<td></td>
<td>Values cannot be modified</td>
</tr>
<tr>
<td></td>
<td>Modifiable does not apply</td>
</tr>
</tbody>
</table>

The default for all SQL data types is that the index column is modifiable.

4 Click File ➤ Save or before changing pages within Table Editor.
Foreign Keys Tasks

➢ To add a foreign key

Note that at least one table in the database must have a primary key or you cannot add a foreign key.

1. Ensure that the “Foreign Keys” page of Table Editor is active. If required, perform the steps for To work with foreign keys.

2. Click Add.

3. Type the name that you want for the new foreign key.

   **Tip** For a list of database object lengths and invalid characters, see Identifier Restrictions by Identifier Type in Advanced Operations Guide.

4. Click OK.

   The new foreign key appears in the Foreign Keys list and the Foreign Keys Details display.

5. Click for “Select Primary Table” to display the list of tables permissible as primary tables.
Click the desired table in the list (only tables with a primary key appear in the list).

The primary field(s) in the table appear in “Primary Table Fields” column.

Match fields in the foreign table with fields in the primary table:
Click the empty cell in the “Foreign Table Fields” column for the corresponding field in the “Primary Table Fields,” then click to displays the list of permissible fields.

Note The data type and size of the fields must match. The list of Foreign Table Fields contains only fields that are the same data type and size as the primary table field being matched.

Click the desired field in the list.
Repeat steps 7 and 8 to match each field listed in the “Primary Table Fields” column with a field in the “Foreign Table Fields” column.
10 Click the desired referential integrity rule: **Delete Restrict** or **Delete Cascade**.

Pervasive PSQL allows a circular delete cascade on a table that references itself. Because of this, use delete cascade with caution. See **Delete Restrict** and **Delete Cascade**, both in Advanced Operations Guide.

11 Click **File** ➤ **Save** or [ ] before changing pages within Table Editor.

➢ **To modify a foreign key**

1 Ensure that the “Foreign Keys” page of Table Editor is active. If required, perform the steps for To work with foreign keys.

2 Click on the desired foreign key in the Foreign Keys list.

3 The Foreign Keys Details displays.

4 As desired, select the primary table, match foreign table fields to primary table fields, and set the referential integrity rule.

See steps 5 through 10 in To add a foreign key.

5 Click **File** ➤ **Save** or [ ] before changing pages within Table Editor.

➢ **To delete a foreign key**

1 Ensure that the “Foreign Keys” page of Table Editor is active. If required, perform the steps for To work with foreign keys.

2 Click on the desired foreign key in the Foreign Keys list.

3 Click **Delete**.

4 Click **Yes** to confirm the deletion.

5 Click **File** ➤ **Save** or [ ] before changing pages within Table Editor.

**SQL View Tasks**

➢ **To copy SQL statements**

1 Ensure that the “SQL View” page of Table Editor is active. If required, perform the steps for To view SQL statements applicable to the table.
Table Editor

2 Position the cursor in the desired statement view: **CREATE Statement** or **ALTER Statement**.

3 With the mouse, select the desired text. (Press and hold the right mouse button then “drag” across the desired lines.)

![Tip]
You can press **Ctrl+A** to select all of the text.

4 Right-click then click **Copy** (or press **Ctrl+C**).

➢ **To maximize or restore view of SQL statements**

1 Ensure that the “SQL View” page of Table Editor is active. If required, perform the steps for **To view SQL statements applicable to the table**.

2 For the desired statement view, **CREATE Statement** or **ALTER Statement**, click the icon in the upper right corner of the view:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Maximize Icon]</td>
<td>Maximizes the statement view.</td>
</tr>
<tr>
<td>![Restore Icon]</td>
<td>Restores the statement view to its size prior to maximizing.</td>
</tr>
</tbody>
</table>
SQL Editor

A Tour of the SQL Editor

The topics in this chapter include:

- SQL Editor Concepts
- Working with Common SQL Objects
- SQL Editor Used in SQL View Tab of Table Editor
- SQL Editor Tasks
SQL Editor Concepts

This section contains the following topics:

- SQL Editor Concepts
- Working with Common SQL Objects
- SQL Editor Used in SQL View Tab of Table Editor
- SQL Editor Tasks

Overview

SQL Editor is one of the editors within Pervasive PSQL Control Center (PCC). The editor allows you to run Structured Query Language (SQL) statements against a Pervasive PSQL database. With SQL statements, you may retrieve, create, change, or delete data in a database provided you have the proper database permissions to perform these actions.

The SQL statements that you may use with SQL Editor are documented in SQL Engine Reference. See especially Grammar Statements.
Caution  Backup all your data definition files (DDFs) and data files before you perform functions through SQL Editor. This tool gives you the ability to modify your database table definitions and data. You could inadvertently change your files in an irreversible manner. Full recovery is possible if you have performed a backup.

Statement Separators
SQL Editor requires a way to differentiate where one statement ends and another begins. The way to differentiate statements is to place a statement separator at the end of each statement. SQL Editor accepts only the pound sign (#) and the semicolon (;) as statement separators. See To select an SQL statement separator.

If you are not using temporary tables, you may use either separator solely or mix their usage within a set of SQL statements. That is, some statements can end with a pound sign and others with a semicolon if you so choose.

Temporary tables begin with “#” or “##.” If you use temporary tables, the pound sign may not be used as a statement separator. Instead, set the statement separator to the semicolon.

Restrictions
The following actions and SQL statements are not supported in SQL Editor:

- Creating a database
- Use of dynamic parameters (indicated by a question mark)
- COMMIT and START TRANSACTION

Displaying Statement Results
SQL Editor displays the results of running SQL statements in the following PCC window views:

- Grid
- Text

In addition, by default, an Outline window view displays a list of the SQL statements in SQL Editor, typically with a shorter line length. For example, you may want the Outline view to show only the first 5 words of each SQL statement in SQL Editor. To set preferences for SQL Editor, click SQL Editor.
Grid Window View

PCC provides a command to execute an individual SQL statement in SQL Editor so that the results appear in the Grid view. The command is called **Execute in Grid** and can be invoked from the **SQL** menu, from a toolbar button, or from within the Outline view.

**Note** The Grid is also used by Table Editor to show table data when you start Table Editor. That is, when you right-click on a table then click Edit. See **To view table data**.

Identifying the Grid Window

The Grid window view, or Grid for short, shows the columns and data in a table. The Grid uses a matrix format, like a spreadsheet, to show the result of running SQL SELECT statements (statements that return data).

Figure 26  Grid Window View

Modifying Data and Adding Rows

The Grid allows you to directly change database data by changing the values in the grid cells. You can also add rows to your table with the Grid. You must have the proper table permissions to affect the database data.

See **To set preferences for PCC Window Views, Grid Tasks, and Assigning Permissions Tasks**.
Records Affected and Scrolling

The Grid caches results locally and initially retrieves 100 records. The Grid displays as many records as its vertical size permits.

As you scroll the vertical scroll bar, more records are retrieved and made available to the Grid. The number of records retrieved appears in the lower right corner of the main window.

Once you scroll to the bottom, the rows fetched indicator reports the total number of records returned by the SQL statement.

Text Window View

PCC provides a command to execute an individual SQL statement in SQL Editor so that the results appear in the Text view. The command is called **Execute in Text** and can be invoked from the SQL menu or from a toolbar button.

The Text window view is automatically used for the results of any SQL statement that is not a SELECT statement. For example, suppose you want to delete some records and have typed a DELETE statement into SQL Editor. If you select the menu command **Execute in Grid**, SQL Editor returns the results to the Text window view, not to the Grid window view.

A command also exists to execute all statements sequentially in the SQL Editor. The command is called **Execute All SQL Statements** and can be invoked from the SQL menu, from a toolbar button, or from within the Outline view. Results from this command always display in the Text view regardless of the statements in SQL Editor.

Identifying the Text Window

The Text window view shows in a text format the result of running SQL statements. You cannot change the data values in the database by changing the text, but you can copy the text.

You may use the Text window view for SELECT statements to show data returned. The data returned appears in a columnar format with each field represented as an underlined heading. The data appears as rows below the headings.
For how to change the font used by the Text window, see To set preferences for Text Output, click Text Output.

For PCC running on an operating system set to an English language locale, the system selects a default font. For non-English locales, PCC seeks to match the “default font” or “system font” if such can be found. Otherwise it will select reasonable font.

![Figure 27 Text Window View](image)

If execution stops because of an error, the Text window view lists the statement that was last run. Knowing the last statement run can help you troubleshoot problems.

**Scrolling and Positioning**

As a convenience, the Text window view automatically scrolls to the top line of the data returned by the last statement executed. For example, suppose that you execute the following two statements sequentially in SQL Editor, each time sending the results to the Text window view: `SELECT * FROM Class` and `SELECT * FROM Billing`.

The Text window view automatically scrolls to the top of the data returned by `SELECT * FROM Billing`, the last statement executed.
Outline Window View

The Outline window view allows you to view the SQL statements in a tree structure. The root node of the tree is the same name as the name of the SQL Editor session to which the outline corresponds.

The number of words displayed in the Outline window view depends on your preferences setting. To set preferences for SQL Editor, click SQL Editor.

You can also execute statements from the Outline window view. The Outline view allows you to select multiple statements to execute (with Ctrl+click). For example, if your Outline view shows three statements as in the figure above, you may choose to execute statements 1 and 3 but not 2. See To run SQL statements in Outline view.
Working with Common SQL Objects

Some SQL objects are dealt with commonly. As a convenience for you, PCC shows the following objects in Pervasive PSQL Explorer and provides commands for their creation and editing:

- Triggers
- Stored procedures
- User-defined functions
- Views

When you use commands to create one of these objects, SQL Editor provides SQL syntax for that object to help you get started. For example, if you choose to create a new view, SQL Editor contains the syntax `CREATE VIEW <viewname> AS`.

Tip You can hover the mouse cursor on a SQL statement to obtain a tool tip on the syntax, which also includes an example.

SQL Editor provides the newly created object with a default name of `object_n`, where `object` is the name of the object and `n` is an integer that starts with one and increments by one. For example, if you create a new view, SQL Editor contains a new tab named “View_1.” After you save the object with a name of your choice, the tab reflects the saved name.
The following table defines the common objects and refers you to SQL Engine Reference for additional information.

Table 14  Description of Common SQL Objects in PCC

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triggers</td>
<td>A type of stored procedure that are automatically executed when data in a</td>
<td>CREATE TRIGGER in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td>table is modified with an INSERT, UPDATE, or DELETE.</td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>Stored procedures</td>
<td>A collection of one or more SQL statements that can take and return user-supplied parameters.</td>
<td>CREATE PROCEDURE in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>User-defined functions</td>
<td>A scalar routine that returns a value.</td>
<td>CREATE FUNCTION in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
<tr>
<td>Views</td>
<td>A database object that stores a query and behaves like a table.</td>
<td>CREATE VIEW in SQL Engine Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Common SQL Object Tasks</td>
</tr>
</tbody>
</table>
SQL Editor

---

**SQL Editor Used in SQL View Tab of Table Editor**

SQL Editor is also used in the SQL View Page of Table Editor. See [SQL View Page](#).
SQL Editor Tasks

This section explains the tasks that you perform with SQL Editor. The tasks are divided into the following categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Tasks</td>
<td>Orient you to the overall use of SQL Editor</td>
</tr>
<tr>
<td>Execution Tasks</td>
<td>Apply to the running of SQL statements</td>
</tr>
<tr>
<td>Grid Tasks</td>
<td>Apply to using the Grid</td>
</tr>
<tr>
<td>Text View Tasks</td>
<td>Apply to using the Text window</td>
</tr>
<tr>
<td>Outline View Tasks</td>
<td>Apply to using the Outline window</td>
</tr>
<tr>
<td>Common SQL Object Tasks</td>
<td>Apply to triggers, stored procedures, user-defined functions, and views</td>
</tr>
</tbody>
</table>

**Note** If you use SQL Editor to change the structure of a table with SQL statements, refresh the Pervasive PSQL Explorer to see the change. Right-click on the **Tables** node in Pervasive PSQL Explorer then click **Refresh**.

**General Tasks**

General tasks orient you to the overall use of SQL Editor.

- To start SQL Editor for a new SQL query
- To start SQL Editor by displaying all records in a table
- To set database context for an SQL query
- To identify editor settings for SQL Editor
- To create an SQL query or script
- To open an SQL script
- To type comments into SQL Editor
- To cancel (undo) or restore (redo) typing actions in SQL Editor
SQL Editor

- To find text or replace text in SQL Editor
- To select text in SQL Editor
- To perform basic editing functions in SQL Editor

**Execution Tasks**
Execution tasks apply to the running of SQL statements.
- To enable the execution commands and icons for SQL statements
- To run a single SQL statement in SQL Editor
- To run selected SQL statements in SQL Editor
- To run all SQL statements in SQL Editor
- To run SQL statements in Outline view

**Grid Tasks**
Grid tasks apply to working with the Grid window.
- To change data within the Grid
- To add rows of data to the Grid
- To delete row(s) of data from the Grid
- To enter a date, time, or timestamp data type in the Grid using scalar functions
- To refresh data in the Grid
- To copy data from the Grid

**Text View Tasks**
Text view tasks apply to working with the Text window.
- To clear results from Text view
- To select and copy text from Text view

**Outline View Tasks**
Outline view tasks apply to working with the Outline window.
- To minimize, maximize, or restore Outline view size

**Common SQL Object Tasks**
Common SQL Object tasks apply to working with triggers, stored procedures, user-defined functions, and views.
- To create a common SQL object
- To modify a common SQL object
- To delete a common SQL object
General Tasks

To start SQL Editor for a new SQL query

1 Start PCC if it is not already running. (See Starting PCC on Windows.)

2 Click File › New › SQL Document or click .
   The Select Database dialog box appears.

3 Click the database in the list for which you want the SQL document to apply, or ensure that the option None is check marked if the SQL document does not apply to a specific database.

   Note The option None is selected by default if an object other than a database, or any of the nodes subordinate to a database, is selected in Pervasive PSQL Explorer.

   Note that the commands to execute SQL statements are disabled if None is specified as the context. See To set database context for an SQL query.

   Select the Set selected database as default for this session option to use the selected database whenever you open a new SQL Editor tab. If you leave this option unselected, you will be prompted to select a database each time you open a new SQL Editor tab.

4 Click OK.

SQL Editor appears as a new window view in PCC. By default, PCC names the new document SQLDoc, where n is an integer that starts with 1 and increments by 1. The document name appears in the tab for SQL Editor.
To start SQL Editor by displaying all records in a table

1. Start PCC if it is not already running. (See Starting PCC on Windows.)
2. Expand the Engines and Databases nodes in Pervasive PSQL Explorer.
3. For the desired database, expand the Tables node.
4. Double-click the table for which you want to see all records (or right-click the table then click Open).
   By default, PCC open SQL Editor and executes a SELECT * FROM statement for the table. Note that the SELECT statement can fail depending on user and column-level permissions.

To set database context for an SQL query

The commands to execute SQL statements are disabled until a database is specified as the context to which the SQL statement applies.

1. Start PCC if it is not already running. (See Starting PCC on Windows.)
2. Perform one of the following actions:
   a. For a new SQL statement, click File ➤ New ➤ SQL Document or click .
      The Select Database dialog appears. Click the database in the list for which you want the SQL document to apply and click OK.
   b. If SQL Editor already contains SQL statements not associated with a database, click .
      The Select Database dialog appears. Click the database in the list for which you want the SQL document to apply.
3 Click **OK**.

**Note** Select the **Set selected database as default for this session** option to use the selected database whenever you open a new SQL Editor tab. If you leave this option unselected, you will be prompted to select a database each time you open a new SQL Editor tab.

To unselect the default database:
Access the Select Database dialog by clicking , and unselect the **Set selected database as default for this session** option.

or

Click **Windows > Preferences** then click the **General** node. Unselect **Do not prompt for new database each time a SQL document is opened**.

The selected database is not maintained across PCC sessions. If you close and reopen PCC, you will have to select a new default database context.

---

**To identify editor settings for SQL Editor**

1 Ensure that the cursor is positioned in SQL Editor.

2 Observe the information blocks along the bottom of the PCC window.
To create an SQL query or script

By default, when you start SQL Editor, you may type in SQL statements. A script is one or more SQL statements saved as a text file.

1. Perform the steps for To start SQL Editor for a new SQL query.
2. Type the SQL statements into SQL Editor.
   Separate SQL statements with a delimiter. You can use the pound sign (#) or the semicolon (;).
3. Optionally, click File ➤ Save As to save the SQL statements as a text file.

To open an SQL script

A script is one or more SQL statements saved as a text file. You can execute the statements in SQL Editor after you open a script in the editor.

1. Click File ➤ Open.
SQL Editor Tasks

2 Navigate to the location of the text file, select the file, then click Open.

By default, the Open dialog looks for files in the PVSW\bin directory with a file name extension of “SQL.”

3 Execute the file. See To run all SQL statements in SQL Editor and To run SQL statements in Outline view.

➢ To select an SQL statement separator

1 On the PCC Window menu, click Preferences. Expand the Pervasive node if it is not already expanded.

2 Click SQL Editor.

3 Select the desired choices for SQL Statement Separator.

Note Based on the separator option you select, PCC looks for the selected character(s) and identifies each as the end of a statement. It sends each identified statement to the database engine and displays results of that statement before sending the next statement.

If you use # as a separator in a script but do not select the #(Pound) option, you will receive an error message when you run the script.

If you do not select ;(Semicolon) as a separator, but use a semicolon as a separator in a script anyway, you will not receive an error message if the statements are properly parsed. This is because the database engine recognizes semicolons as separators. However, PCC will not display the results for all the statements. It will only display results for one statement (probably the first statement). As far as PCC is concerned, if you don’t select a semicolon as a separator, statements separated by a semicolon are a single statement.
To type comments into SQL Editor

Single-line comments are indicated by double dashes (--) or double slashes (//--). Each comment must be on a separate new line or after the statement separator on an existing line.

SQL Editor also supports the use of a start/end comment block that can span multiple lines (/* */).

1. Click at the beginning of the line where you want a comment.
2. Type "--" or "//" followed by your comment text.

The following example shows valid comments.

```
SELECT * FROM t1#
-- This is a valid comment
// and so is this
SELECT * FROM t2# -- This is valid after the # sign
```

The following example shows multi-line comments.

```
SELECT * FROM t1# -- single line comment
/* This is a comment block that spans two lines.
Statements inside this block are ignored */
SELECT * FROM t2#
```

To cancel (undo) or restore (redo) typing actions in SQL Editor

1. Perform one of the following actions:
   
   a. Click Edit ▶ Undo (or press Ctrl+Z) to cancel typing actions.
   b. Click Edit ▶ Redo (or press Ctrl+Y) to restore typing actions.

To find text or replace text in SQL Editor

1. Click Edit ▶ Find/Replace (or press Ctrl+F).

A dialog appears on which you specify a text string to find or replace.
SQL Editor Tasks

➢ To select text in SQL Editor
1. Perform one of the following actions:
   a. Click Edit ➔ Select All to select all of the contents of SQL Editor.
   b. Press and hold down the left mouse button and drag the cursor across the text you want to select.

➢ To perform basic editing functions in SQL Editor
1. Click Edit then click the function you want: cut, copy, paste, and so forth.

Statement Execution Tasks

➢ To enable the execution commands and icons for SQL statements
1. Follow the steps for To set database context for an SQL query.

➢ To run a single SQL statement in SQL Editor
1. Position the cursor on the statement or select the statement.
2. Perform one of the following actions:
   a. Click SQL ➔ Execute in Grid or SQL ➔ Execute in Text.
   b. Press F9 or Shift+F9.
   c. Click or .

Note SQL Editor automatically uses the Text window view for the results of SQL statements that are not SELECT statements. Only SELECT statements use the Grid window view.
SQL Editor

➢ To run selected SQL statements in SQL Editor

1 Press and hold down the left mouse button and drag the cursor across the statement that you want to run.

You may select one or more statements.

2 Perform one of the following actions:
   a. Click SQL › Execute in Text or SQL › Execute All SQL Statements.
   c. Click or .

Note SQL Editor automatically uses the Text window view for the results of SQL statements that are not SELECT statements. Only SELECT statements use the Grid window view.

➢ To run all SQL statements in SQL Editor

1 Click SQL › Execute All SQL Statements, press F10, or click .

Ensure that either no statements are selected or that all statements are selected. If you select a portion of the statements in SQL Editor, only the selected portion executes.

➢ To run SQL statements in Outline view

1 To execute all statements in Outline view, right-click on the root node then click Execute All Statements.

   To execute one or more statements, click on the desired statement(s).

   Note that you can select multiple statement by using Ctrl+click. The statements do not have to be contiguous.

2 If multiple statements are selected, right-click on one of the selected statements, then click Execute Selected Statements.
If a single statement is selected, perform one of the following actions:

a. Right-click on the statement, then click **Execute in Grid** or **Execute in Text**.

b. Press F9 or Shift+F9.

c. Click or .

---

**Note** SQL Editor automatically uses the Text view for the results of SQL statements that are not SELECT statements. Only SELECT statements use the Grid.

---

**Grid Tasks**

➢ **To change data within the Grid**

1. Click the Grid cell that contains the value you want to change.

   **Tip** By default, the entire contents becomes selected when you click the cell. Press Delete or Backspace to delete the entire contents of the cell.

2. Change the data in the cell.

3. Move the cursor outside of the cell (for instance, press Tab or click outside of the cell).

   **Caution** Moving the cursor from the cell automatically saves the data changes to physical storage. You cannot explicitly save the changes made to the cell.
To add rows of data to the Grid

1. Click on the Grid.

The Add Rows dialog appears. For example, the following image shows the dialog for the “Billing” table that is part of the sample database DEMODATA.

2. Click in the Value cell for each Column Name and type the desired value.

The value must be a data type valid for that column.

Tip: You can copy data from Grid cells and paste it into the Value cells. Click on a Grid cell then right-click. Click Copy. Click on a Value cell on the Add Rows dialog then right-click. Click Paste. Also note that Ctrl+C and Ctrl+V provide the copy and paste actions, respectively.

3. Click Add.

The record is added to the table. Also note that the option Refresh Grid on Exit becomes enabled.

If you want to add multiple records, you can change values for specific value cells then click Add. If you want to clear all of the value cells, click Reset.
4 Optionally, click **Refresh Grid on Exit** if you want the table data refreshed.

> ![Refresh Grid on Exit](image)

When you close the Add Rows dialog, a refresh re-executes the statement last executed in SQL Editor.

5 Click **Close**.

If **Refresh Grid on Exit** is enabled, the Grid displays the record(s) that you just added (assuming that the last statement executed in SQL Editor was SELECT * FROM Billing).

> ![Grid Example](image)

### To delete row(s) of data from the Grid

**Caution** Deleting a row from the Grid removes that record from physical storage. No undo feature is available to reclaim the deleted record.

1 Click any cell within the row (the record) that you want to delete.

You may also select and delete multiple rows. To select multiple rows, press and hold down the Shift or Ctrl key, then click a cell in each desired row.

2 Click **Edit ▶ Delete** or click ![Delete Button](image).

3 Click **OK** to confirm the deletion.
To enter a date, time, or timestamp data type in the Grid using scalar functions

1 As a convenience, you can type the following scalar functions for date, time, and timestamp in Grid cells:

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Scalar Function¹ As Typed in Grid Cell</th>
</tr>
</thead>
</table>
| Date        | • now()  
             • curdate()  
             See also NOW() and CURDATE(), both in SQL Engine Reference.                                      |
| Time        | • now()  
             • curtime()  
             See also NOW() and CURTIME() in SQL Engine Reference.                                            |
| Timestamp   | • now()  
             See also NOW() in SQL Engine Reference.                                                            |

¹ The names are case insensitive. NOW() and now() are equivalent. The parentheses are required. That is, NOW is invalid but NOW() is valid.

**Note** You can also omit the seconds for a time data type provided that you include “AM” or “PM.” For example, 10:30 AM is a valid entry. Time defaults to “AM” if you omit “AM” or “PM.” For example, 10:30:00 is entered as 10:30:00 AM.

To refresh data in the Grid

1 Click on the Grid.

A refresh re-executes the statement last executed in SQL Editor and sends the results to the Grid.
SQL Editor Tasks

➢ To copy data from the Grid
1 Perform one of the following actions:
   a. To select the data for an individual cell, click in the cell, then click (or right-click and click Copy). By default, the entire content of the cell is selected.
   b. To select an entire row, right-click on any cell then click Copy Text or click .
      You may also select multiple rows. To select multiple rows, press and hold down the Shift or Ctrl key, then click a cell in each desired row.

   Note When you copy an entire row or multiple rows, the rows are pasted in the same layout as they appear in the Text window view. You can specify the number of characters between the pasted columns. Click Window then expand the Pervasive node in the Preferences tree. Click Text Output in the tree and set the desired value for Number of spaces between columns.

Text Window Tasks

➢ To clear results from Text view
1 Click on the Text view.

➢ To select and copy text from Text view
1 Perform one of the following actions:
   a. Press and hold down the left mouse button and drag the cursor across the text you want to select.
   b. Right-click within the Text view then click Select All.
2 Right-click then click Copy.
Outline View Tasks
See also To run SQL statements in Outline view.

➢ To minimize, maximize, or restore Outline view size
1 Click the desired sizing button:

<table>
<thead>
<tr>
<th>Button</th>
<th>Sizing Action</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Minimizes the Outline window</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Maximizes the Outline window</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Restores the Outline window to its size before it was minimized</td>
</tr>
</tbody>
</table>

Common SQL Objects Tasks
Common SQL objects include triggers, stored procedures, user-defined functions, and views.

➢ To create a common SQL object
1 In PCC Pervasive PSQL Explorer, expand the Engines node and the registered server nodes to display the available databases.
2 Right-click on the database for which you want the common SQL object to apply (or right-click on any of the database's subordinate nodes).
3 Click New then one of the following depending on the object that you want to create:
   • Function
   • Stored procedure
   • Trigger
   • View
A new SQL Editor is opened that contains a default name for the object on a tab. The name is in the form object_n, where object is the name of the object and n is an integer that starts with one and increments by one. For example, if you create a new view, a new SQL Editor contains a tab named “View_1.” After you save the object with a name of your choice, the tab reflects the saved name.

4 Modify the skeletal SQL statement for the common object.

Tip Hover the mouse cursor on the statement to obtain a tool tip on the syntax, which also includes an example.

5 Click File » Save or .

➢ To modify a common SQL object

1 In PCC Pervasive PSQL Explorer, click the database in the list for which you want to modify the common SQL object.

2 Expand the node for the object (function, stored procedures, user-defined functions, or triggers) that you want to modify.

3 Double-click the object that you want to modify or right-click the object then click Edit.

A new SQL Editor is opened that contains a tab. The tab name reflects the name by which you saved the object.

➢ To delete a common SQL object

1 In PCC Pervasive PSQL Explorer, click the database in the list for which you want to delete the common SQL object.

2 Expand the node for the object (function, stored procedures, user-defined functions, or triggers) that you want to delete.

3 Click the object that you want to delete, then perform one of the following actions:
   a. Right-click, then click Delete.
   b. Press Delete.
   c. Click on Pervasive PSQL Explorer.

Note that you can select multiple objects for deletion by using Ctrl+click or Shift+click.
Pervasive System Analyzer (PSA)

Usage Information for the Diagnostic Utility PSA

The following are the sections found in this chapter:

- PSA Concepts
- PSA GUI Visual Reference
- PSA Tasks
PSA Concepts

Pervasive System Analyzer (PSA) is a utility that allows you to perform the following actions:

- View the Pervasive components on your system with version information, usage status, size, and location.
- Identify duplicate components on your system.
- Test your network communications to verify connectivity.
- Test the transactional interface to verify connectivity to the database engine.
- Test the relational interface to verify connectivity to the database engine.

View Modules

This option allows you to view all Pervasive components and any other files you specify. You can add additional components to the search list. Their versions, usage status, file size, and location is displayed. Duplicate files are identified so you can resolve any potential conflicts.

View modules scans a machine for Pervasive components to determine which ones are loaded into memory.

Test Active Installation

This option allows tests of the network connectivity between a Pervasive PSQL client and the database engine, and tests the functionality of the transactional and relational interfaces. If errors are detected, PSA gives you detailed troubleshooting information to help you resolve the problem.

Network

The Network Communication tests verify that your client or workstation can communicate with network protocols to reach the machine on which the Pervasive PSQL database engine is installed. For the TCP/IP protocols, both IPv4 and IPv6 numeric and named addresses are supported. See also Drive-based Formats in Getting Started With Pervasive PSQL for additional details about IPv6.
Using **Advanced Settings**, you can select the protocols to test as well as the number of stress test messages to send. By default, the network test connects using any available protocol that is installed on the system and configured for use in Pervasive PSQL. See **Supported Protocols** configuration parameter found in Advanced Operations Guide.

The progress bar will complete for all selected tests. The steps involved in testing network communication are:

1. Verify Available Protocols
2. Verify Network Client Availability
3. Qualify Target Name
4. Resolve Target Location to network address
5. Verify Server Address
6. Verify Server Connection
7. Run Stress Test

**Transactional Engine**

This test verifies the ability of your client to connect to the Pervasive PSQL database through the transactional interface (Btrieve).

When you run this test, PSA attempts to perform basic database operations that are common to most transactional interface applications. If your machine passes this test, then the following is verified:

- The Pervasive PSQL transactional interface is responding
- Your client interface components are installed correctly
- The network communication between the client and the database engine is functioning correctly
- Transactional interface applications running on your computer should function correctly

**Relational Engine**

This test verifies the ability of your client to connect to the Pervasive PSQL database engine through the relational interface (SQL). When you run this test, PSA attempts to perform common SQL database operations. A dialog box displays during the tests to show progress.

If your machine passes this test, then the following is verified:
Pervasive System Analyzer (PSA)

- Your Pervasive PSQL relational engine is running
- Your client interface components are installed correctly
- The network communication between the client and the database engine is functioning correctly
- SQL applications running on your computer should function correctly

**Frequently Asked Questions**

Listed below are some frequently asked questions about PSA.

- **What is the default log file name?**
- **Can I use a different log file name?**
- **What is the local default location for the log file?**
- **What is the remote default location for the log file?**
- **What kind of information is contained in the log file?**
- **What happens to the information in the log file each time the utility is run?**
- **How do I run PSA?**
- **When would I want to use PSA?**

**What is the default log file name?**

The default file name for the log file is **PSA.log**.

**Can I use a different log file name?**

You can rename the file, but the name applies only for that session of PSA. PSA defaults the name to PSA.log the next time you execute PSA.

**What is the local default location for the log file?**

The default local location for the log file is under the application data directory in the logs folder.

For default locations of Pervasive PSQL files, see Where are the Pervasive PSQL files installed? in Getting Started With Pervasive PSQL.

**What is the remote default location for the log file?**

The default location for the log file is always the directory where psawizrd.exe is located on a Windows machine.
What kind of information is contained in the log file?
The log file records any process performed by PSA and includes the associated timestamp.

What happens to the information in the log file each time the utility is run?
New information is appended to the log file each time PSA runs if the option Append to log file is selected (see Figure 29). If the option is cleared, the log file contains only information for that particular execution of PSA.

How do I run PSA?
See To start PSA.

When would I want to use PSA?
The following lists the most common situations where you would want to use PSA:

- You are encountering network errors and wish to test your client's connectivity to a machine running a Pervasive PSQL database engine.
- Your Pervasive-based application is not functioning correctly and you wish to test the functionality of the transactional or relational interface components to the database engine.
- You wish to view the Pervasive PSQL components on the system, loaded in memory, or both.
PSA GUI Visual Reference

The following dialog in PSA provides access to its functionality. Click on an area of the figure to learn more about the item.

Figure 29  PSA Main Dialog

<table>
<thead>
<tr>
<th>GUI Object</th>
<th>Description</th>
<th>Related Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>View loaded Pervasive modules</td>
<td>Displays all current Pervasive PSQL components and version information in a table. Also allows you to add additional components to the matrix.</td>
<td>View Modules Tasks</td>
</tr>
<tr>
<td>Test active installation</td>
<td>Test the Pervasive PSQL installation in three ways: network connectivity, functionality of the transactional interface, and functionality of the relational interface. Select the options that correspond to the tests you wish to perform.</td>
<td>Test Active Installation Tasks</td>
</tr>
<tr>
<td>Test network</td>
<td>Test the network connectivity to a machine running the database engine. The network tests display detailed information about any problems encountered during the test and provide suggestions on how to remedy the problems. If you do not have Internet Explorer installed, this test is not available because the test uses an embedded HTML component to display results.</td>
<td>To test your network</td>
</tr>
<tr>
<td>Test transactional engine</td>
<td>Tests the functionality of the transactional interface (Btrieve) to the database engine.</td>
<td>To test the transactional interface</td>
</tr>
<tr>
<td>GUI Object</td>
<td>Description</td>
<td>Related Information</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Test relational</td>
<td>Tests the functionality of the relational interface (SQL) to the database</td>
<td>To test the relational interface</td>
</tr>
<tr>
<td>engine</td>
<td>engine.</td>
<td></td>
</tr>
<tr>
<td>Log file</td>
<td>Allows you to specify a different log file location than the default. PSA</td>
<td>To specify a different location for the</td>
</tr>
<tr>
<td></td>
<td>logs detailed information on the tests it performs. You can use this log</td>
<td>PSA log file</td>
</tr>
<tr>
<td></td>
<td>file to review tests at a later time or to forward them to Pervasive</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Technical Support for further review.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The <strong>Append to the log file</strong> option adds content to the end of the log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>file to provide a running history. If this option is cleared, the log file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>starts anew and only information from the current PSA session is captured.</td>
<td></td>
</tr>
</tbody>
</table>
Pervasive System Analyzer (PSA)

PSA Tasks

General Tasks
- To start PSA

View Modules Tasks
- To select options for View Modules

Test Active Installation Tasks
- To test your network
- To test the transactional interface
- To test the relational interface

Log Files Tasks
- To specify a different location for the PSA log file
- To view the log file at the completion of PSA

General Tasks ➤ To start PSA
1. Access Pervasive System Analyzer from the operating system Start menu or Apps screen or execute psawizrd.exe from a command prompt.

For default locations of Pervasive PSQL files, see Where are the Pervasive PSQL files installed? in Getting Started With Pervasive PSQL.

View Modules Tasks ➤ To select options for View Modules
1. Start PSA and click Next.
2. Click View Loaded Pervasive Modules.
3 Click **Next**.

A dialog displays similar to the following figure.

Figure 30  View Modules Section of PSA

If **In Memory** is selected, PSA searches for all Pervasive components loaded in memory, regardless of whether they are located in the PATH or other specified locations.

If you want to add components to the search list, do the following:

a. Click **Additional modules** (if it is not already selected).

b. Type the filename (without a drive letter or path) in the text box.

c. Click **Add Component**.

d. If the component you added is not in the path, add the path to the searched locations.

If you want to add more paths to the search locations, do the following:

a. Click **Additional locations**.

b. Click **Add Location**.

c. Browse to the location using the directory selector and select the desired location.

d. Click **OK**.

e. If you also want all directories below the location to be searched, click **Include subfolders**.

4 Click **Next**.
5 View the grid that shows the components found. If files of the same name are detected, the multiple occurrences are marked with an icon on the left side.

If you see multiple occurrences of a component, you can adjust the list so that the occurrences sort together. To do this, click the Module column heading.

Multiple occurrences of a file does not necessarily represent a problem with your configuration. Their identification can help you troubleshoot issues with components depending on the situation. For example, if you see two Pervasive components both marked with the same version, you may want to check which is loaded in memory and that the components are located where you expect.

6 Click Next.

7 Click Finish if you are finished using PSA, or click View Log File to view the log.

Test Active Installation Tasks

- To test your network
  1 Start PSA and click Next.
  2 On the Options dialog, click Test active installation (if it is not already selected).
  3 Click Test network communication (if it is not already selected).
  4 Click Next.
  5 For Target machine, type the machine name or TCP/IP address of the machine you wish to test. You may also type "localhost" for the local machine. The machine can be the machine where you are currently located or a remote machine.
6 If you want to control how the tests are run, click **Advanced Settings** and complete the following steps:

   a. By default, PSA uses the first available protocol. You can force PSA to use one or more protocols by clicking **Use Only These Protocols**.
   
   b. Select the protocol options that you want PSA to test. Note that the NetBIOS protocol is not supported on Pervasive PSQL Server. The SPX protocol is not supported on Pervasive PSQL Workgroup.
   
   c. Specify the number of stress test messages to send by typing a value for **test messages**. The default is 75. (After successfully connecting using any protocol, PSA sends stress test messages to ensure that the connectivity is functioning correctly.)
   
   d. Click **OK** to save the changes.

7 Click **Next** to start the test.

   PSA runs a series of tests and displays the results.

   If the test succeeds, PSA informs you that all test messages were successfully transmitted. If the test fails, PSA lists the issues along with tips on how to resolve them. The tips are also written to the PSA log file.

8 Click **Next**.

9 Click **Finish** if you are finished using PSA, or click **View Log File** to view the log.

   ➤ **To test the transactional interface**

1 Start PSA and click **Next**.

2 On the Options dialog, click **Test active installation** (if it is not already selected).

3 Click **Test transactional engine** (if it is not already selected).

   This test runs a series of transactional operations to simulate your client requester interacting with the database engine.

4 Click **Next**.
Pervasive System Analyzer (PSA)

5 Provide a path name to the samples directory on the machine running the database engine. The default path should be correct if the database engine is running on the local machine. You can type a path, or browse to one if you click ![image].

6 Select the operations you want to perform for the transactional interface (by default, all are selected):
   - Create Data File (write access required)
   - Read Data File
   - Update Data (write access required)
   - Insert Data (write access required)

7 Click **Next**.

PSA performs the tests and displays the results. A check mark indicates a test succeeds and an "x" indicates a test fails.

If all of the tests succeed, your client requester can use the transactional interface to communicate with the database engine.

8 Click **Next**.

9 Click **Finish** if you are finished using PSA, or click **View Log File** to view the log.

➢ **To test the relational interface**

1 Start PSA and click **Next**.

2 On the Options dialog, click **Test active installation** (if it is not already selected).

3 Click **Test relational engine** (if it is not already selected).

   This test runs a series of SQL operations to simulate your client requester interacting with the database engine.

4 Click **Next**.

5 For **Machine Name**, type the name or IP address of the machine where the engine data source name (DSN) resides, or browse to the machine (click ![image]).

   The name “localhost” is valid if you are testing the local machine.
PSA Tasks

6 For **Engine DSN**, type the name of the DSN for the data source you want to test. The default, demodata, uses the DSN for the sample database installed with the database engine.

7 Select the operations you want to perform for the relational interface (by default, all are selected):
   - Create Table
   - Read Data
   - Update Data
   - Insert Data

8 Click **Next**.

PSA performs the tests and displays the results. A check mark indicates a test succeeds and an "x" indicates a test fails.

If all of the tests succeed, your client requester can use the relational interface to communicate with the database engine.

9 Click **Next**.

10 Click **Finish** if you are finished using PSA, or click **View Log File** to view the log.

Log Files Tasks

➢ To specify a different location for the PSA log file

1 Start PSA and click **Next**.

2 In the Log File field, type a path to the PSA log file, or browse to the desired location (click **»**).

   If you want a log file that contains information only about the current PSA session, clear the **Append to log** file option.

   You can also specify a different name for the log file, but the name applies only for that session of PSA. PSA defaults the name to PSALog.txt the next time you execute PSA.

➢ To view the log file at the completion of PSA

   Click **View Log File** to display the PSA log file.

   A summary of the tasks PSA completed is listed for you.
Pervasive System Analyzer (PSA)
This chapter discusses command line interface (CLI) utilities available for Pervasive PSQL.

The chapter contains the following sections:

- CLI Utilities Overview
- Command Line Interface Utility Reference
CLI Utilities Overview

In addition to providing GUI utilities, Pervasive PSQL provides a number of command line interface (CLI) utilities that you can use. In most cases, these utilities duplicate functionality you can perform with GUI utilities.

For default locations of Pervasive PSQL files, see Where are the Pervasive PSQL files installed? in Getting Started With Pervasive PSQL.

Platforms that Include CLI Utilities

These utilities are provided in the following installations:

- Windows - Server, Workgroup, and a limited set on the Client
- Linux - Server, and a limited set on the Client

The summary of utilities in the section that follows notes which utilities are present in a server or client install.

Where to Find CLI Utilities

Please note the location based on your platform.

Windows

In Windows, the utilities are installed to the BIN directory of your Pervasive PSQL installation. If you installed to the default installation location, your utilities are located in file_path\PSQL\bin. Since the Pervasive PSQL installation places your install directory in the PATH, these utilities should be available from any command prompt.

Linux

In Linux, utilities are installed to /usr/local/psql/bin. The user psql has the necessary environment variables to use the utilities. If you wish to use utilities from accounts other than psql, follow these instructions in Getting Started With Pervasive PSQL: Pervasive PSQL Account Management on Linux.

Utilities by Platform and Engine Type

The following tables outlines the command line utilities, the platform on which they are made available, as well as if a graphical user interface is available for the utility.
<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
<th>GUI Available</th>
<th>Windows</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcfg</td>
<td>Configures Pervasive components</td>
<td>Pervasive PSQL Control Center (see Configuration Reference in Advanced Operations Guide)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>bdu</td>
<td>Imports data into a database</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>brmon</td>
<td>Monitors Pervasive PSQL activity</td>
<td>Monitor Utility (see Monitoring Database Resources in Advanced Operations Guide)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>btadmin</td>
<td>Creates and administers database user names and passwords</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>butil</td>
<td>Repairs and manipulates data files</td>
<td>Maintenance Utility (see Manipulating Btrieve Data Files with Maintenance in Advanced Operations Guide)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>cllicadm</td>
<td>Applies and administers user licenses.</td>
<td>License Administrator (see License Administration)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>w64cllicadm</td>
<td>(Windows 64-bit)</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>cllicadm64</td>
<td>(Linux 64-bit)</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>clipaadm</td>
<td>Phone Authorization Utility used to authorize product keys when internet connection is unavailable. (Phone Authorization Only)</td>
<td>Pervasive Phone Authorization Wizard</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>w64clipaadm</td>
<td>(Windows 64-bit)</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>clipaadm64</td>
<td>(Linux 64-bit)</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>dbmaint</td>
<td>Creates and administers named databases</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>dsnadd</td>
<td>Creates and administers Engine DSNs on the server</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Command Line Interface Utilities

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
<th>GUI Available</th>
<th>Windows</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>isql</td>
<td>Allows you to run SQL statements interactively and test connectivity to a DSN.</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>isql64</td>
<td>(Linux 64-bit)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>licgetauth</td>
<td>Transmits Authorization Request Data and retrieves Authorization Key Data. (Offline Authorization Only)</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>w64licgetauth</td>
<td>(Windows 64-bit)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>licgetauth64</td>
<td>(Linux 64-bit)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notification Viewer</td>
<td>Notification Viewer provides two interfaces: system tray icons and a graphical user interface (GUI). Although not strictly a CLI utility, it is mentioned here for reference. See Pervasive Notification Viewer.</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>psc</td>
<td>Manipulates Pervasive PSQL services</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>psregedit</td>
<td>Configures Pervasive components by editing the Pervasive registry</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>psregsvr</td>
<td>Register Pervasive components</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>pvdbpass</td>
<td>Specifies user names and passwords for secure databases</td>
<td>No</td>
<td>Pervasive PSQL Control Center (see Pervasive PSQL Security in Advanced Operations Guide)</td>
<td>Yes</td>
</tr>
<tr>
<td>pvddl</td>
<td>Processes SQL statements in a command file</td>
<td>Yes</td>
<td>Pervasive PSQL Control Center (see SQL Editor)</td>
<td>Yes</td>
</tr>
<tr>
<td>pvmdconv</td>
<td>Converts V1 metadata to V2 metadata</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Utility* | *Description* | *GUI Available* | *Windows* | *Linux* |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Server</td>
<td>Client</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Server</td>
<td>Client</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

246
## CLI Utilities Overview

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
<th>GUI Available</th>
<th>Windows</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Server</td>
<td>Client</td>
</tr>
<tr>
<td>pvnetpass</td>
<td>Specifies user names and passwords for remote servers</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>rbldcli</td>
<td>Rebuilds MicroKernel data files.</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Command Line Interface Utility Reference

This section provides a reference for the following command line interface utilities:

- `bcfg`
- `bdu`
- `bmon`
- `btadmin`
- `butil`
- `clilcadm`
  - `w64clilcadm`
  - `clilcadm64`
- `clipaadm`
  - `w64clipaadm`
  - `clipaadm64`
- `dbmaint`
- `dsnadd`
- `isql`
  - `isql64`
- `licgetauth`
  - `w64licgetauth`
  - `licgetauth64`
- `psc`
- `psregnsvr`
- `pvdbpass`
- `pvdsl`
- `pvmdconv`
- `pvnetpass`
- `rbldcli`
The `bcfg` utility is documented in Advanced Operations Guide. See Configuration Through CLI Utility.
**bdu**

**Description**

The Bulk Data Utility (BDU) is a command line utility that allows you to load data from a delimited text file into a Pervasive PSQL table. The table and database must already exist.

The BDU, the table, the database, and the Pervasive PSQL database engine must all be located on the same machine. The delimited text file must be locally accessible by the database engine server through a local drive, mapped drive, mounted folder, or shared folder.

You may use a default delimiter or a user-specified delimiter. The delimiting character must not be contained in the data itself. The following tables list the permissible delimiters.

<table>
<thead>
<tr>
<th>Delimiter</th>
<th>Indicated By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab</td>
<td>\t (default)</td>
</tr>
<tr>
<td>Any single printable character (control characters are not printable, except null, tab, new line, and carriage return)</td>
<td>(\t, A, t, l, and so forth)</td>
</tr>
</tbody>
</table>

**Note** Pervasive PSQL does not support the use of NULL terminator (\0) or double quote (") as column delimiters.

<table>
<thead>
<tr>
<th>Delimiter</th>
<th>Indicated By</th>
</tr>
</thead>
<tbody>
<tr>
<td>New line character</td>
<td>\n (default)</td>
</tr>
<tr>
<td>Carriage return</td>
<td>\r</td>
</tr>
<tr>
<td>Carriage return line feed (CR LF)</td>
<td>\r\n</td>
</tr>
</tbody>
</table>

The BDU supports only the single quote (') and the double quote (") characters as text qualifiers. The data file may contain column values enclosed by single quotes or by double quotes. For example, the following column values are enclosed by double quotes and delimited by the TAB character: "Fred\t22\t2459\tSales"\t.
The BDU treats consecutive column delimiters as NULL values. If the utility finds consecutive column delimiters, it inserts a NULL value into the column, provided the column is nullable.

No qualifiers are allowed for a NULL value. The following column data a NULL value in the second column. Note that qualifiers are not included for that column: "Fred\"\t"2459\"\t"Sales\"\t.

**Synopsis**

```bash
bdu {database_name} {table_name} {data_file}

[-e max_errors]
[-r reject_file]
[-f first_row]
[-l last_row]
[-t field_term]
[-n row_term]
[-o output_file]
[{-u login_id} {-p password}]
[-q text_qualifier]
[-h]
```

**Note** When loading data with BDU into a secured database for which the Btrieve Security policy is set to "Mixed," the supplied credentials (user name and password) must match those of a Pervasive PSQL database user account and an operating system user account.

**Options**

Table 17  Bulk Data Utility Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory/Optional</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>database_name</td>
<td>Mandatory</td>
<td></td>
<td>Database name to connect to the local Pervasive PSQL engine</td>
</tr>
<tr>
<td>table_name</td>
<td>Mandatory</td>
<td></td>
<td>Name of the table to be populated</td>
</tr>
<tr>
<td>data_file</td>
<td>Mandatory</td>
<td></td>
<td>Name and location of the delimited text file</td>
</tr>
</tbody>
</table>
You are not required to change any Pervasive PSQL configuration settings to use BDU.

BDU loads data into a table using the accelerated mode. During the load of data, the MicroKernel does not perform transaction logging.
If you use archival logging, back up your data files again

**Error Logging**

By default, BDU logs all information and error messages to the standard error stream (stderr). You may specify a log file to which the utility writes the information or error messages.

Two types of errors are not logged: critical and recoverable. With critical errors, BDU exits because it cannot perform error recovery. For example, a missing delimited data file is a critical error.

With recoverable errors, BDU skips the error and continues processing. The utility keeps a count of such skipped errors and exits when it reaches a user-specified threshold. By default, the threshold is set to zero.

**Constraints**

The following constraints apply to loading data with BDU.

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any Referential Integrity (RI) error is considered an RI violation</td>
<td>Row is rejected</td>
</tr>
<tr>
<td>Any unique or primary key violations</td>
<td>Row is rejected</td>
</tr>
<tr>
<td>No value specified for a non-NULL column¹</td>
<td>Row is rejected regardless of column’s default value</td>
</tr>
<tr>
<td>No value specified for a nullable column¹</td>
<td>NULL inserted regardless of column’s default value</td>
</tr>
<tr>
<td>Table into which data is being loaded contains insert triggers</td>
<td>BDU returns an error and does not attempt to load the table. Drop the insert triggers on the table then re-run BDU</td>
</tr>
<tr>
<td>Table into which data is being loaded contains CLOB or BLOB columns</td>
<td>BDU does not attempt to load the table</td>
</tr>
<tr>
<td>Order of rows</td>
<td>BDU treats the delimited data file as unordered. The original order of rows may not be preserved.</td>
</tr>
<tr>
<td>Date fields</td>
<td>The only supported format is yyyy-mm-dd</td>
</tr>
<tr>
<td>Time fields</td>
<td>The only supported format is HH:MM:SS</td>
</tr>
</tbody>
</table>
Command Line Interface Utilities

Best Practices
If possible, run BDU when the database load is minimal or when no concurrent sessions exist on the table being loaded.

If the table being loaded contains any indexes, drop the indexes before using BDU. Re-create the indexes after the load is complete.

If the table being loaded contains any columns with check constraints, drop the check constraints before using BDU. Re-specify the constraints after the load is complete.

Sample Source File
The following content may be used to create a sample delimited text file. You may use the file to verify the usage examples. The examples refer to the sample file as data_file.txt.

Note that, because the following content is comma delimited, you must specify the -t parameter (-t,) with BDU. The -t parameter is required for any delimiter except the TAB character.

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp fields</td>
<td>The only supported format is yyyy-mm-dd HH:MM:SS.MS</td>
</tr>
</tbody>
</table>

1 BDU is not aware of default values for a column defined during table creation or update

Examples
The following examples assume that a table named BDU_Table is part of the Demodata sample database. To add such a table to Demodata, use the following query:

```
CREATE TABLE BDU_Table (Name CHAR(20) NOT NULL CASE,
                         PhoneNo INTEGER, BuildingName CHAR(25) NOT NULL CASE,
                         RoomNo UINT NOT NULL, HeadOfDept UBIGINT NOT NULL)
```
To run BDU with the default options:

```bash
bdu demodata BDU_Table C:\data_file.txt
```

**Note** The input data must be TAB delimited to use default options. If the input data is not TAB delimited, you must specify the delimiter with the `-t` parameter.

For example, to use the data from Examples, which is comma delimited, you would run BDU as follows:

```bash
bdu demodata BDU_Table C:\data_file.txt -t ,
```

To run the BDU for a database that requires username and password:

```bash
bdu demodata BDU_Table C:\data_file.txt -u <username> -p <password>
```

To run the BDU with max errors option:

```bash
bdu demodata BDU_Table C:\data_file.txt -e <no of errors user wants to allow>
```

For instance, for loading to continue until 100 errors have occurred:

```bash
bdu demodata BDU_Table C:\data_file.txt -e 100
```

To run the BDU with a specific column delimiter option:

```bash
bdu demodata BDU_Table C:\data_file.txt -t <column delimiter>
```

Example:

When the source file contains text in which each row is separated by ,

```bash
bdu demodata BDU_Table C:\data_file.txt -t ,
```

To run the BDU with a specific row delimiter option:

```bash
bdu demodata BDU_Table C:\data_file.txt -n <row delimiter>
```

For instance, when the source file contains text in which each row is separated by \n:
bdu demodata BDU_Table C:\data_file.txt -n 

To run the BDU with a specific start row option:
bdu demodata BDU_Table C:\data_file.txt -f <line no. from which user wants loading to begin>

To run the BDU with a specific end row option:
bdu demodata BDU_Table C:\data_file.txt -l <line no. at which user wants loading to end>

You may combine parameters. To load the first 15 rows from the source file containing data that is separated by | and is enclosed in ':
bdu demodata BDU_Table C:\data_file.txt -f 1 -l 15 -t |
bmon

The bmon utility is documented in Advanced Operations Guide. See Monitor Command Line Interface.
Command Line Interface Utilities

**btadmin**

**Description**
The btadmin utility is used to create and update the flat file btpasswd, which stores user names and passwords for authentication of Pervasive PSQL users. Users given administrator rights can monitor engine status and configure the engine remotely.

**Synopsis**
```
badmin [ -p password] [a+] [a-] [-r] username
```

**Options**
- **-p** Specify the password. If this option is not specified, you will be prompted to enter the password.
- **a+** Gives administrator rights for this user.
- **a-** Removes administrator rights for this user.
- **-r** Remove user name from btpasswd file.

**username**
Creates or updates the username in the btpasswd file. If username does not exist in this file, an entry is added. If it does exist, the password is changed.

**See Also**
butil(1)

**Notes**
To administer the engine from a remote workstation, you must supply a user name and password. Upon initial installation of Pervasive PSQL v11 SP3, the supplied default is admin with an empty password.

Use btadmin to add more administrators:
```
% btadmin [-p password] [a+] username
```

258
This utility creates a record in `btpasswd` for user `username` with password `password` (if option `-p` is not used, then you will be asked to enter a password). If a user already exists, then his password is changed as specified.

By default a user is created without administration permissions. You can use the `a+` option to give administration rights to the user. You can remove this right by using `a-`.

To remove a user record from the password file, enter:

```
% btadmin -r username
```

Every time the `btpasswd` file is changed, the previous version is backed up to `btpasswd-`. 
**butil**

**Description**

The Pervasive PSQL Maintenance Utility, or butil, is a command line utility that performs command file and data manipulations on a data file.

The maintenance utility performs the following file and data manipulations:

- Starts and stops continuous operations for use in performing server backups.
- Recovers changes made to a file between the time of the last backup and a system failure.
- Imports and exports ASCII, unformatted, and SDF sequential data.
- Copies data between data files.
- Returns MKDE version information.

Continuous operation is an MKDE feature that enables you to back up files while they are in use by Pervasive PSQL-based applications. Two maintenance commands, `startbu` and `endbu`, begin and end continuous operation on a file or set of files.

**Synopsis**

```
butil
-clone outputFile sourceFile [/O<owner | *>] [/pagecompress | pagecompressoff] [/recordcompress | recordcompressoff] [/UIDuname /PWDpword [/DBdbname]]
-clrownwer sourceFile /O<owner | *> [/UIDuname /PWDpword [/DBdbname]]
@endbu sourceFile | @listFile>
```

```
@commandFile [commandOutputFile]
-copy sourceFile outputFile
   [/O< owner1 | *>] [/O<owner2 | *>] [/UIDuname /PWDpword [/DBdbname]]
-create outputFile descriptionFile [< Y | N >] [/UIDuname /PWDpword [/DBdbname]]
-drop sourceFile < keyNumber | SYSKEY >
   [/O<owner | *>] [/UIDuname /PWDpword [/DBdbname]]
-endbu < /A | sourceFile | @listFile > [/UIDuname /PWDpword [/DBdbname]]
-index sourceFile indexFile descriptionFile
   [ /O< owner | *>] [/UIDuname /PWDpword [/DBdbname]]
-load unformattedFile outputFile [/O<owner | *>] [/UIDuname /PWDpword [/DBdbname]]
-recover sourceFile unformattedFile [/O<owner | *>] [/UIDuname /PWDpword [/DBdbname]]
```
Command Line Interface Utility Reference

- rollfwd <sourceFile | volume | drive | @listFile>
  
  
  [/E<keyLength>] [/H] [/V] [/O<ownerList | owner | *]]
  
  [/A] [/UIDname /PWDpword [/DBdbname]]

- save sourceFile unformattedFile
  
  [Y indexFile | N <keyNumber | -1>] [/O<owner1 | *>
  
  [/O<owner2 | *>]] [/UIDname /PWDpword [/DBdbname]]

- setowner sourceFile /O<owner | *> level [/UIDname /PWDpword
  
  [/DBdbname]]

- sindex sourceFile <descriptionFile | SYSKEY> [keyNumber] [/O<owner
  
  | *>] [/UIDname /PWDpword [/DBdbname]]

- startbu <sourceFile | @listFile> [/UIDname /PWDpword [/DBdbname]]

- stat <sourceFile | /O<owner | *>] [/UIDname /PWDpword [/DBdbname]]

- ver

**Note** On Linux distributions, all “/” parameters use the hyphen (“-“) instead of the slash. For example, the /O parameter for butil -copy is -O, as in butil -copy -O.

**Options**

Maintenance Utility command options are not case sensitive unless the option is a filename.

If you run butil without specifying a command option or with an invalid command option, a usage message is printed. The usage message indicates that there is an optional /S command line argument to butil. This argument is ignored under Linux.

For a complete discussion of the utility commands, options, and examples, see the section Btrieve Command-Line Maintenance Utility (butil) in Advanced Operations Guide.

**See Also**

syslogd in Linux man pages

Btrieve API Guide, which describes the API for the transactional interface.
clilcadm

**Description**  The command line License Administrator utility manages the user count licenses on your engine. The Windows and Linux 32-bit command line utility is named clilcadm, the Windows the 64-bit utility is named w64clilcadm, and the Linux 64-bit version of this utility is named cliadm64.

**Note**  On Linux, this utility can only be run by user accounts belonging to group `pvsw`. See Getting Started With Pervasive PSQL for information on Pervasive PSQL Linux utilities and user accounts.

**Synopsis**

```
clilcadm -a <key> | -c [key] [force] | -d <key> | -g <key> <filename> | -h | -i <key> | -n [key] | -s <servername> | -t | -u <username> | -p <password>
```

```
w64clilcadm -a <key> | -c [key] [force] | -d <key> | -g <key> <filename> | -h | -i <key> | -n [key] | -s <servername> | -t | -u <username> | -p <password>
```

```
cliadm64 -a <key> | -c [key] [force] | -d <key> | -g <key> <filename> | -h | -i <key> | -n [key] | -s <servername> | -t | -u <username> | -p <password>
```

**Options**  For a complete discussion of the command line options, see License Administrator CLI Syntax.

**See Also**

licgetauth

License Administration documents License Administration and the associated utilities in detail.
clipaadm

Description
The clipaadm utility allows you to authorize Pervasive PSQL via the telephone, in the event that it is not possible to authorize keys online, remotely, or offline.

With this utility, options are not passed but instead, responses are given to prompts the utility displays in order to gather information required to complete product telephone authorization.

Note Phone authorization is not available for Pervasive PSQL Vx Server.

Synopsis
clipaadm

Options
This section details the prompts displayed and information needed to complete telephone authorization.

1. Please enter the 30 character product key.
   Enter the product key you want to authorize.

2. Repeat the authorization request code to phone support
   If you obtained your product key from Pervasive Software, call Pervasive Software support and give them the Authorization Request Code displayed in this step.
   If you did not obtain your product key from Pervasive Software, contact the vendor or original equipment manufacturer (OEM) from whom you obtained the key. Provide the vendor or OEM personnel the Authorization Request Code displayed in this step.

3. Enter the authorization key provided by support: xxxx-xxxx-xxxx-xxxx-xxxx-xxxx-xxxx-xxxx
   Enter the Authorization Key provided by Pervasive support, or by the vendor or OEM personnel, to authorize Pervasive PSQL.

See Also
License Administration documents License Administration in detail.
**dbmaint**

**Description**

The dbmaint utility manages named databases.

---

**Note** This utility can only be run by user accounts belonging to group pvsw. See Getting Started With Pervasive PSQL for information on Pervasive PSQL Linux utilities and user accounts.

---

**Synopsis**


- **add new database name**
  - `a` -n Dbname [-b] [-i] [-e] [-l ddictpath] [-d datapath]
- **delete database name**
  - `d` -n Dbname
- **list database names**
  - `l` [-a]. The -a option displays the full information about the dbnames.
- **modify database name security policy**
  - `m` -n Dbname -s securitymode

**Options**

**Commands**

- **add, a** Add database name
- **del, d** Delete database name
- **list, l** List database names
Options

- b 
  Create bound database

- c=codepage
  Set the database code page. Zero specifies the server default (-c=0).

- i
  Create database with relational integrity

- e
  Do not create dictionary files for database

- nDBName
  Specify database name

- ldictpath
  Specify dictionary path

- ddatapath
  Specify datapath

- a
  Show detail information about dbnames in database list

- ssecuritymode
  Specify Btrieve security policy for database. Valid choices are: Classic, Mixed, Database

Examples

To create a database named “TEST” with relational integrity:

% dbmaint a -i -nTEST

Note Unless a datapath is specified, the new database will be in the default location, $PVSW_ROOT/data. Likewise, if a dictionary path is not specified, the dictionary will be created in the default location.

To delete the same database:

% dbmaint d -nTEST

To create a database named “mydbase” with a database code page of CP932:

% dbmaint a -nmydbase -c=CP932

For the same database, to set the code page to the default operating system code page:

% dbmaint m -nmydbase -c=0

To see a list of valid code pages (specify an invalid code page and dbmaint returns a list of valid ones):
% dbmaint m -nmydbase -c=xyz

Dbmaint returns something similar to the following:

Bad code page "xyz" should be: ASCII, ISO8859_1, CP437, CP1252, UTF-8, CP1250, CP1251, CP1253, CP1254, CP1255, CP1256, CP1257, CP1258, CP737, CP775, CP850, CP852, CP855, CP857, CP858, CP862, CP866, CP932, or EUCJP

To list all database names with full information:

% dbmaint l -a

To modify the security policy of the DefaultDB database to Mixed:

% dbmaint m -nDefaultDB -sMixed

**See Also**

dsnadd, butil(1), badmin(1), syslogd(1), smb.conf(5)

Data Encoding in Getting Started With Pervasive PSQL.
dsnadd

**Description**

Dsnadd simplifies the setup of a new ODBC data source to connect to a Pervasive PSQL database. It modifies the odbc.ini file by providing the appropriate properties for the new data source.

Pervasive PSQL follows the UNIXODBC standard by using the odbcinst.ini file in /usr/local/psql/etc to specify a 32-bit and a 64-bit ODBC driver. DSNs that reference the “Pervasive ODBC Interface” driver description point to the odbcinst.ini information from the odbc.ini file. A single DSN can be used by both a 32-bit application and a 64-bit application. See also Notes.

Optionally, dsnadd provides options that let you create legacy-style DSNs that specify a 32-bit driver name in odbc.ini rather than pointing to odbcinst.ini. However, such DSNs are not accessible to 64-bit applications.

**Synopsis**

- To create a DSN on the server that connects to a named database:

  dsnadd -dsn=myDSN -db=DBname

  This creates a DSN with the description “Pervasive ODBC Interface” that can be used by both a 32-bit application and a 64-bit application on the server. The DSN is not visible as an “Engine DSN” in ODBC Administrator running on a Windows client.

- To add a DSN on a client that connects to a named database on a server:

  dsnadd -dsn=myDSN -db=DBname -host=psqlhost

  This creates a DSN with the description “Pervasive ODBC Interface” that can be used by both a 32-bit application and a 64-bit application on the client.

- To add a deprecated, legacy-style Client DSN with the description “Pervasive ODBC Client Interface,” execute the following command:

  dsnadd -dsn=myDSN -sdsn=engineDSN -host=psqlhost -clntdsn

  Note that the DSN is accessible only to 32-bit applications.

- To add a deprecated, legacy-style Engine DSN on the server with the description “Pervasive ODBC Engine Interface:”
Command Line Interface Utilities

dsnadd -dsn=myDSN -db=DBname -engdsn

Note that the DSN is accessible only to 32-bit applications.

- To list existing DSNs:
  dsnadd -l

Where:
myDSN is a name you want to assign to the new data source.
DBname is the name of the named database on the Pervasive PSQL host.
psqlhost is the name of the host where your Pervasive PSQL Server is installed.
engineDSN is the name of the Engine DSN on the Pervasive PSQL host.

Options

The following options are the most commonly used:

-\db= Name of the database (local or remote) to which the DSN is associated

-\dsn-name= | -dsn= The Data Source Name

-help Display syntax and options help for the dsnadd utility

-\l List existing DSNs

openmode=<0|1|-1|-4> | -omode=<-0|1|-1|-4> Specify the default file open mode for files opened with the current connection. The default is 0, or “Normal.” Can be used only with local connections, not remote client connections. For more information on file open modes, see DSN Open Mode in SQL Engine Reference.

-srv-host= | -host= Server host name

-srv-port= | -port= Server port number. The default is 1583. (See also Changing the Default Communication Ports in Getting Started With Pervasive PSQL.)

-translate=< none | auto > Encoding translation to use for character data. The default is “none,” meaning that no character data is translated between the client and server on the assumption that the client and server use the same operating system encoding.

See None and Automatic, both in SQL Engine Reference.

The following options are used to create deprecated, legacy-style DSNs:
Command Line Interface Utility Reference

- clntdsn
  Create a deprecated, legacy-style Client DSN with the driver description
  "Pervasive ODBC Client Interface."

  Rather than using Client DSNs, new applications or revised 32-bit
  applications should create a DSN that connects to a named database using
  -db=.

- engdsn
  Create a deprecated, legacy-style Engine DSN with the driver description
  "Pervasive ODBC Engine Interface."

  Rather than using Engine DSNs, new applications or revised 32-bit
  applications should create a DSN that connects to a named database using
  -db=.

- sdsn=
  For use with deprecated, legacy-style Client DSNs. Name of an Engine DSN
  on the Pervasive PSQL Server.

The following options are typically used only for application development and testing:

- dsn-desc= | -desc=
  dsnadd provides a default DNS description in odbc.ini. If you want to override
  the default description, use this option to specify a descriptive string of your
  choosing. If the descriptive string contains the space character, you must
  quote the entire string with double quotes.

- drv-desc=
  dsnadd handles the driver descriptions based on the absence or presence of
  the -db, -clntdsn and -engdsn options. If specified, must be one of the
  following:
  • Pervasive ODBC Interface
  • Pervasive ODBC Client Interface
  • Pervasive ODBC Engine Interface

- drv-path= | -drv=
  The path where the driver libraries are located. The default is /usr/local/psql/
  lib:$HOME/lib.

- odbc-ini= | -ini=
  ODBC.ini file name (for example, /usr/local/psql/etc/odbc.ini)

Examples

The following example creates a server-side DSN named “acctingdb” that connects to a local database named “region1accting.”

    dsnadd -dsn=acctingdb -db=region1accting

The following example creates a client-side DSN named “USInvoices” that connects to a database named “DomesticOrders” on a remote server named “USInventory.”

    dsnadd -dsn=USInvoices -db=DomesticOrders
    -host=USInventory

The following example creates a deprecated, legacy-style Client DSN named “bkordersclnt” that references an Engine DSN named
Command Line Interface Utilities

“backorderssrv” on a machine named “JapanSvr2” and uses automatic encoding.

dsnadd -dsn=bkordersclnt -sdsn=backorderssrv -
    host=JapanSvr2 -translate=auto -clntdsn

The following example creates a deprecated, legacy-style Engine DSN named “partscatalog” that connects to a database named “partscatalog.”

dsnadd -dsn=partscatalog -db=partscatalog -engdsn

Notes

On Linux distributions, individual ODBC drivers are loaded through the driver manager UNIXODBC. The driver manager maintains a mapping from Data Source Names (DSNs) to the specific Pervasive PSQL ODBC drivers.

The installation of Pervasive PSQL Server 64-bit or Client 64-bit leaves user-defined, pre-existing 32-bit DSNs as is, meaning they are not immediately accessible from a 64-bit application. For new DSNs, the installation of both products assigns a 32-bit and a 64-bit ODBC driver in odbcinst.ini. This assignment allows a single DSN to be used by both a 32-bit application and a 64-bit application.

If you want pre-existing 32-bit DSNs to be accessible to both 32-bit and 64-bit applications, you must recreate them as DSNs that connect to a named database.

ODBC and Data Source Names (DSNs)

The application bitness does not have to match the bitness of the Pervasive PSQL Server product. For example, the 64-bit ODBC driver or the 32-bit ODBC driver can be used to connect to either Pervasive PSQL Server 64-bit or Pervasive PSQL Server 32-bit.

For the Pervasive PSQL Client, however, the application bitness does have to match the bitness of the Pervasive PSQL Client on the client machine. That is, if you want to use a 64-bit application on the client, the Pervasive PSQL 64-bit Client must be installed.
Pervasive PSQL allows DSNs to specify three different ODBC driver descriptions, as explained in the following table.

Table 18  Pervasive PSQL ODBC Driver Descriptions for Linux

<table>
<thead>
<tr>
<th>ODBC Driver Description in INI Files</th>
<th>Driver Library Installed With</th>
<th>Behavior for All Products Installed With</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pervasive ODBC Engine Interface</td>
<td>Pervasive SQL Server 64-bit</td>
<td>• Installation assigns an ODBC driver in odbc.ini for compatibility with pre-existing Engine DSNs</td>
</tr>
<tr>
<td></td>
<td>Pervasive SQL Server 32-bit</td>
<td>• Connects to a local named database</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• No longer created by default with dsnadd (the <code>engdsn</code> option must be specified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Visible when viewing “Engine DSNs” in ODBC Administrator running on a Windows client</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For use by 32-bit applications that are already coded to use Engine DSNs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 32-bit Engine DSNs are deprecated on Linux.</td>
</tr>
<tr>
<td>Pervasive ODBC Client Interface</td>
<td>Pervasive SQL Server 64-bit</td>
<td>• Installation assigns an ODBC driver in odbc.ini for compatibility with pre-existing Client DSNs</td>
</tr>
<tr>
<td></td>
<td>Pervasive SQL Server 32-bit</td>
<td>• Connects to a local or remote named database or an Engine DSN</td>
</tr>
<tr>
<td></td>
<td>Pervasive SQL Client 32-bit</td>
<td>• For use only by 32-bit applications that are already coded to use Client DSNs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• No longer created by default when -host is specified with dsnadd (the <code>clntdsn</code> option must be specified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 32-bit Client DSNs are deprecated on Linux.</td>
</tr>
<tr>
<td>Pervasive ODBC Interface</td>
<td>Pervasive SQL Server 64-bit</td>
<td>• Installation assigns ODBC drivers in odbcinst.ini for use by new 32-bit and 64-bit DSNs</td>
</tr>
<tr>
<td></td>
<td>Pervasive SQL Client 64-bit</td>
<td>• Connects to a local or remote named database</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The recommended driver description to use for 32-bit and 64-bit applications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Created by default with dsnadd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Not visible as “Engine DSNs” in ODBC Administrator running on a Windows client.</td>
</tr>
</tbody>
</table>
Frequently Asked Questions

The following table answers some frequently asked questions (FAQs) about ODBC and DSN support for Linux.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| What do I need to do about DSNs if I port my 32-bit application to 64-bit? | If the application uses DSN-less connections that connect using “Pervasive ODBC Client Interface,” change the ODBC driver description to “Pervasive ODBC Interface.”
If the application uses DSNs, you must create new DSNs that connect to a named database. |
| What is a so-called “DSN-less” connection?                               | A DSN-less connection is one that connects to a named database using the ODBC driver “Pervasive ODBC Client Interface” (for pre-existing 32-bit applications) or the “Pervasive ODBC Interface” driver (for 32-bit or 64-bit applications). |
| Can I still create deprecated, legacy-style Engine DSNs and Client DSNs? | Yes, but you must specify the -engdsn or -clntdsn option with dsnadd. DSNs created with either option support only 32-bit applications. |
| If I am using ODBC Administrator on a Windows client, why do I not see my DSNs? | On 64-bit Windows operating systems, 64-bit system DSNs are distinct from 32-bit system DSNs because of the registry design. If you are using the 64-bit ODBC Administrator, you will not see the 32-bit system DSNs, and vice versa. |
| What if my application uses DTI to manage DSNs?                          | The DTI functions for DSNs manage only 32-bit Engine DSNs. Therefore, the DTI functions for DSNs are deprecated along with the 32-bit Engine Interface ODBC driver.  
An alternative to using DTI to manage DSNs is to use SQL and ODBC statements. For example, you could use CREATE DATABASE to create a named database and SQLConfigDatasource to configure the DSNs. |
| Going forward, is there a recommended strategy for ODBC connections?     | Yes. New applications or revised 32-bit applications, local or remote, should connect to a named database.                           |

ODBC Header Files

The sql.h, sqltypes.h, and sqlext.h header files for ODBC contain differences for the compilation of 32-bit and 64-bit applications. Refer to the ODBC documentation on the UNIXODBC Web site for a discussion of 64-bit ODBC. For example, you may find the following information useful: http://www.unixodbc.org/doc/ODBC64.html.
See Also  btadmin, dbmaint, isql
isql

**Description**

isql is an interactive ODBC test utility that you can use to test your DSNs for their connectivity to databases and to execute SQL statements once you are connected to a database. The 32-bit utility is named isql and is installed with Pervasive PSQL Server 32-bit, Pervasive PSQL Client 32-bit, and Pervasive PSQL 64-bit. The 64-bit utility is named isql64 and is installed with Pervasive PSQL Server 64-bit and with Pervasive PSQL Client 64-bit.

The only difference between the two utilities is the type of DSN to which they can connect. By default, dsnadd creates DSNs that are accessible to both 32-bit and 64-bit applications (the DSNs specify the driver description “Pervasive ODBC Interface”). You can test connectivity of such DSNs with isql or isql64. See Pervasive PSQL ODBC Driver Descriptions for Linux.

Deprecated, legacy-style DSNs are accessible only to 32-bit applications. Therefore, you can test connectivity of such DSNs only with isql. See FAQs About ODBC and DSN Support for Linux.

For example, to connect to the DEMODATA sample database included with Pervasive PSQL, run isql (or isql64) with the DSN as the first parameter: isql DEMODATA or isql64 DEMODATA. (The DSN for DEMODATA specifies the driver description “Pervasive ODBC Interface” so either utility can test its DSN connectivity.)

The utility puts you in an interactive state with the database. From that state, you can query the database (such as `SELECT * FROM Department`).

To enable security on a database using isql, first connect to the database as the “Master” user, then use the SET SECURITY statement in SQL to set the Master user password. For example:

```
isql DEMODATA Master
SET SECURITY = password
```

See SET SECURITY in SQL Engine Reference.

To connect to a secured database, pass the user name and password as the second and third parameters respectively to the isql utility. For example, to connect to DEMODATA as user “Master” using password “vforge,” enter `isql64 DEMODATA Master vforge` or `isql DEMODATA Master vforge`.
**Synopsis**

```
<isql | isql64> DSN [UID [PWD]] [options]
```

Where:

- **DSN** is the data source name for the database to which you want to connect. Always required.
- **UID** is the user name to connect to the Pervasive SQL database engine for a secured database. Required only for a secured database.
- **PWD** is the password for UID. Required only if UID used.

**options** is one or more of the options as defined below.

**Options**

- `-b` Suppress prompts for batch processing. See Notes.
- `-c` Display column names on first row (use with `-d`)
- `-dx` Delimit columns with character `x`.
- `-l locname` Set locale to `locname`.
- `-m n` Limit column display width to `n` characters.
- `-v` Display verbose explanations for errors and warnings.
- `--version` Display version of unixODBC in use.
- `-w` Wrap results in an HTML table.
- `-x0xHH` Delimit columns with `HH`, where `x` is in hex. For example, 0x09 is the tab character.

**Commands**

Once you are in interactive mode, the following commands may be used.

- **sql-statement** A valid SQL statement to execute against the database to which isql or isql64 is connected.
- **help [tablename]** Display column information. The output is the same as for the `dbo.fSQLColumns` catalog function. See `dbo.fSQLColumns` in *SQL Engine Reference*.

If `tablename` is omitted, the output is for all tables in the database, including system tables.

- **quit** Exits interactive mode.
Notes

By default, isql and isql64 display prompt information when in interactive mode (such as “Connected!,” “sql-statement,” “help [tablename],” and “quit”). For redirection and piping of output to a file, you may not want the prompt information. The -b option suppresses the output of the prompt information.

Isql and isql64 support redirection and piping for input and output. In addition, both utilities can process a file containing multiple SQL statements. Each statement must end with a carriage return/line feed. The last line of the file must be a blank line. See Examples.

Examples

The following example connects to an unsecured database named “acctspay” that will be accessed by a 64-bit ODBC application running on a 64-bit client:

isql64 acctspay

The DSN is also named “acctspay” and specifies the ODBC driver description “Pervasive ODBC Interface.”

The following example connects to a secured database named “payroll” as user “Master” with a password of “j77b99:”

isql payrollsecdb Master j77b99

The DSN is named “payrollsecdb” and specifies the deprecated, legacy-style ODBC driver description “Pervasive ODBC Engine Interface.” (You could also use isql to test the DSN if the DSN had specified the ODBC driver description “Pervasive ODBC Interface”.)

The following example shows how to process multiple SQL statements. Suppose that you want to run the following two queries against the DEMODATA sample database:

select count(*) from billing
select count(*) from person

Create a file (named “two-queries.sql” for discussion purposes) with the two lines. Include a blank line as the last line in the file.

Run the following command:

cat two-queries.sql | isql demodata -b

The result is as follows:

+------------+
| EXPR_1     |
+------------+
| 1315       |
SQLRowCount returns 1
1 rows fetched

<table>
<thead>
<tr>
<th>EXPR_1</th>
<th>1500</th>
</tr>
</thead>
</table>

SQLRowCount returns 1
1 rows fetched

Note the use of the -b option to suppress the prompting information from the output. Without the -b option, the result is as follows:

<table>
<thead>
<tr>
<th>Connected!</th>
<th>sql-statement</th>
<th>help [tablename]</th>
<th>quit</th>
</tr>
</thead>
</table>

SQL> +------------+
<table>
<thead>
<tr>
<th>EXPR_1</th>
<th>1315</th>
</tr>
</thead>
</table>

SQLRowCount returns 1
1 rows fetched

SQL> +------------+
<table>
<thead>
<tr>
<th>EXPR_1</th>
<th>1500</th>
</tr>
</thead>
</table>

SQLRowCount returns 1
1 rows fetched

See Also: dsnadd
licgetauth

Description
The licgetauth utility is used in the second phase of the offline authorization process used for authorizing product keys. This utility is used in conjunction with clilcadm to complete the offline authorization process.

Note
Phone authorization is not available for Pervasive PSQL Vx Server.

Synopsis
licgetauth.exe [output_filename.ath]

Options

output_filename.ath The name for the authorization output file.

See Also
clilcadm
Considerations For No Internet Access
psc

Description
Psc stands for Pervasive service controller. The utility retrieves and sets control information about Pervasive PSQL services. You must have administrator authority to run psc.

Synopsis
psc < start | stop | restart | query | getpolicy > servicename
or
psc setpolicy servicename < automatic | manual | disabled >

Options
A service specifies the name of a program, routine, or process that performs a specific system function to support other programs, particularly at a low level (close to the hardware). Servicename specifies the name given to the service key in the registry. Note that service key name may differ—and in most cases does differ—from the service display name.

The options described below are case-insensitive.

start Starts a Pervasive PSQL service running
stop Terminates the running of a Pervasive PSQL service
restart Terminates the running of a Pervasive PSQL service then starts the service running again
query Specifies whether servicename is running or not
getpolicy Retrieves the type of startmode (automatic, manual, or disabled) associated with servicename
setpolicy Sets the type of startmode (automatic, manual, or disabled) associated with servicename
automatic The service starts automatically when the operating system starts
manual The service must be started manually after the operating system starts
disabled The service is disabled and does not start after the operating system starts
**Examples**

To start the Workgroup Engine service manually:

```bash
psc start psqlWGE
```

To start the Cache Engine service manually:

```bash
psc start psqlCE
```

To stop, then restart the Pervasive PSQL transactional and relational services:

```bash
psc restart psql.all
```

Note that “psql.all” is a shortcut method available only with the psc utility. It affects both the “Pervasive.SQL (transactional)” service and the “Pervasive.SQL (relational)” service, and can be used with the psc parameters start, stop, or restart.

**Return Codes**

The `psc` utility returns the following codes (DOS ERRORLEVEL) for the state of the service after the `psc` command has been run.

<table>
<thead>
<tr>
<th>String</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BTI_SERVICE_ACCESS_DENIED</td>
<td>32775</td>
</tr>
<tr>
<td>BTI_SERVICE_ALREADY_RUNNING</td>
<td>32778</td>
</tr>
<tr>
<td>BTI_SERVICE_CANNOT_ACCEPT_CTRL</td>
<td>32779</td>
</tr>
<tr>
<td>BTI_SERVICE_CONTINUE_PENDING</td>
<td>32772</td>
</tr>
<tr>
<td>BTI_SERVICE_DATABASE_LOCKED</td>
<td>32780</td>
</tr>
<tr>
<td>BTI_SERVICE_DEPENDENCY_DELETED</td>
<td>32783</td>
</tr>
<tr>
<td>BTI_SERVICE_DEPENDENCY_FAIL</td>
<td>32784</td>
</tr>
<tr>
<td>BTI_SERVICE_DISABLED</td>
<td>32782</td>
</tr>
<tr>
<td>BTI_SERVICE_DOES_NOT_EXIST</td>
<td>32785</td>
</tr>
<tr>
<td>BTI_SERVICE_DUP_NAME</td>
<td>32776</td>
</tr>
<tr>
<td>BTI_SERVICE_EXISTS</td>
<td>32786</td>
</tr>
<tr>
<td>BTI_SERVICE_INVALID_CTRL</td>
<td>32789</td>
</tr>
<tr>
<td>BTI_SERVICE_INVALID_NAME</td>
<td>32777</td>
</tr>
<tr>
<td>BTI_SERVICE_MARKED_FOR_DELETE</td>
<td>32790</td>
</tr>
<tr>
<td>String</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>BTI_SERVICE_NOT_ACTIVE</td>
<td>32787</td>
</tr>
<tr>
<td>BTI_SERVICE_PAUSE_PENDING</td>
<td>32773</td>
</tr>
<tr>
<td>BTI_SERVICE_PAUSED</td>
<td>32774</td>
</tr>
<tr>
<td>BTI_SERVICE_REQUEST_TIMEOUT</td>
<td>32788</td>
</tr>
<tr>
<td>BTI_SERVICE_RUNNING</td>
<td>32771</td>
</tr>
<tr>
<td>BTI_SERVICE_START_PENDING</td>
<td>32769</td>
</tr>
<tr>
<td>BTI_SERVICE_STOP_PENDING</td>
<td>32770</td>
</tr>
<tr>
<td>BTI_SERVICE_STOPPED</td>
<td>32768</td>
</tr>
</tbody>
</table>
psregedit

Description

psregedit is used to manage the Pervasive Registry on Linux. You must be root user or a member of the group pvsw to make changes to the Pervasive Registry.

Synopsis

psregedit
-key keyname [ -r ]
-key keyname -value valuename
-set -key keyname [-type type] value
-set -key keyname -value valuename [-type type] value
-delete -key keyname
-delete -key keyname -value valuename
-export -key keyname [-file filename]
-import [-file filename]

Where:
keyname is in the form PS_HKEY\Subkey and PS_HKEY is one of the following: PS_HKEY_CONFIG, PS_HKEY_CONFIG_64, PS_HKEY_CLASSES, PS_HKEY_CLASSES_64, or PS_HKEY_USER. Subkey is a subordinate key under a major key.
valuename is the name assigned to the Registry value or “default.”
type is PS_REG_STR, or PS_REG_UINT32, or PS_REG_UINT64.
value is the value assigned to valuename.
filename is the name of a file, which may include a path.

Options

-key Get the key value. If -value is not specified, then the entire key contents are displayed. To view all subkeys, specify -r (for recursive).
-set -key Set the key value. If -value is not specified, the default value will be set. VALUE must be appropriate to the TYPE specified. PS_REG_STR is assumed if -type is not specified.
-delete -key Delete the specified key or value. If -value is not specified, then the entire key and all subkeys are deleted.
-export -key  Export the given key, including all values and subkeys to standard output or to the named file.

-import [-file filename]  Import keys and values from either the standard input or from the named file.
Command Line Interface Utilities

**psregsvr**

*Description*  
psregsvr is used to register components in the Pervasive registry.

*Synopsis*  
```
psregsvr [ -s ] [ -u ] { [ -f file ] | filename }
```

*Options*  
- `-s`  
  Silent. Do not print any status or error messages.
- `-u`  
  Unregister. If not specified, register is assumed.
- `-f file`  
  Specifies a text file with PCOM modules listed one per line.
- **filename**  
  Specifies a single PCOM module to register.
pvdbpass

**Description**

pvdbpass allows users to change their passwords for secure databases without administrator intervention.

**Synopsis**

The utility will prompt for the passwords with this syntax

```
pvdbpass database username [-server name] [-port number]
```

This syntax includes the old and new passwords.

```
pvdbpass database username password newpassword
    [-server name] [-port number]
```

**Options**

- **database**
  Database in which the username is defined (this can be a database name or a server DSN)

- **username**
  The user whose password will be changed.

- **password**
  The current password for the user. You must provide the original password in order to modify it. You can either provide the password as a parameter or omit it and be prompted.

- **newpassword**
  The new password for the user. See Identifier Restrictions by Identifier Type in Advanced Operations Guide for password restrictions.

  Note: If the new password begins with a non-alphabetic character, the password must be enclosed in single quotes. If the existing password begins with a non-alphabetic character, do not enclose it in single quotes (see examples).

- **-server name**
  Optional. Server name on which the database is defined. If you do not specify this option, the local machine is assumed.

- **-port number**
  Optional. TCP port on which the SQL engine running on servername is listening. If you do not specify this option, the default port 1583 is assumed. See also Changing the Default Communication Ports in Getting Started With Pervasive PSQL.

**Examples**

To change the Master user’s password and be prompted:

```
pvdbpass demodata Master
```
Command Line Interface Utilities

To change an existing password to one that does not start with an alphabetic character (use single quotes):

```
pvdbpass demodata Joe oldpassword '123'
```

To change a password on a remote server:

```
pvdbpass demodata Joe oldpass newpass -server finance1
```
pvddl

**Description**

pvddl is used to execute a series of SQL statements in a command file.

**Synopsis**

```
pvddl database commandfile
```

**Options**

- **database**
  
  Database against which the SQL statements in `commandfile` are to be executed (this can be a database name or a server DSN).

- **commandfile**
  
  Text file that contains the SQL statements. Certain categories of SQL statements, such as data definition language, are better suited for use in `commandfile`. (Contrast this with a data manipulation statement such as SELECT. A SELECT statement can be used, but the result set is not returned to standard output.) You need a separator character between each command in your command file. See `-separator character`.

- **-separator character**
  
  Character used in `commandfile` to separate SQL statements. The valid choices are any printable character. However, ensure that `character` does not occur within any of the SQL statements. Common `character` choices include the pound sign (#), semicolon (;), and at sign (@).

- **-username username**
  
  Name of a user defined for a database with security enabled.

- **-password password**
  
  Password for the user identified by `username`.

- **-server servername**
  
  Name of the server on which the database is defined. If you do not specify this option, the local machine is assumed. You may also specify the IP address of the server.

- **-port number**
  
  TCP port number on which the database engine running on `servername` is listening. If you do not specify this option, the default port 1583 is assumed. (Port 1583 is the default port used for the relational interface.)
Command Line Interface Utilities

- **stoponfail**: Stop when the first SQL error is encountered in commandfile. Pvdll returns an error code of PS_E_FAIL if an error is encountered (which equates to -2147467259 decimal). The default action is for pvdll to continue after an SQL error is encountered.

- **log logfile**: Write output to a file instead of to standard output (stdout). Logfile specifies the name of the file to which output is logged and, optionally, a path to the file. If path is omitted, logfile is created in the same directory in which dvdd resides.

**See Also**

SQL Engine Reference for more information about supported SQL syntax.
Pervasive PSQL includes a conversion utility, pvmdconv, to convert V1 metadata to V2 metadata. This command line utility is located in the BIN subdirectory under the installation directory.

**Synopsis**

```
```

**Parameters**

- `-o <1 | 2>` Specifies the output format of the metadata. The choices are 1 or 2, which refer to version 1 (V1) and version 2 (V2) metadata, respectively.
  
  **Note:** The current release of pvmdconv does not support converting DDFs from V2 metadata to V1 metadata.

- `-d path_to_DDFs` Specifies the directory containing the DDFs that you want to convert. The new DDFs are created in the same directory.

- `-n database_name` Specifies the name of the Pervasive PSQL database.

- `[-v]` Specifies to change the metadata version of the database in dbnames.cfg. For example, if `-o=2`, the metadata version of the database is changed to V2 metadata. If the `-v` parameter is used, pvmdconv does not change the metadata version of the DDFs.

- `[-ddf]` Specifies to change the metadata version of the DDFs. If this parameter is used, pvmdconv does not change the metadata version of the database in dbnames.cfg.

- `[-s server_name]` Specifies the name or IP address of the remote server where the database resides. This parameter is required if the pvmdconv utility being executed resides on one machine but the database resides on another. This parameter defaults to “localhost,” so it is not required if pvmdconv and the database reside on the same machine.

- `[-i server_login_name]` Specifies the name required to log in to the remote server where the database resides. `Server_login_name` is a name administered by the operating system.
Description

Pvmdconv allows you to perform the following actions:

- Convert only DDFs from one version of metadata to another version of metadata. The database to which the DDFs belong remains categorized as is. That is, the metadata property for the database, which is stored in a special file named dbnames.cfg, does not change. This action is sometimes referred to as “migrating” the DDFs.

- Set the metadata version of a database to V1 metadata or to V2 metadata. The metadata property of a database, which is stored in a special file named dbnames.cfg, changes. The DDFs for the database retain their metadata version.

  This action requires that the DDFs for each metadata version must already exist in the location path_to_ddfs. That is, you must have already converted (migrated) the DDFs from V1 metadata to V2 metadata.

- Convert the DDFs and set the metadata property of the database in dbnames.cfg. That is, combine the first two actions. This is the default action of pvmdconv.

Having different conversion actions provides more control over how you can convert metadata. For example, you may choose first to convert only the DDFs and check for errors. If no errors occur, you could then update the metadata version in dbnames.cfg.
Conversion
From V1 to V2
Metadata

All of the data from the V1 DDFs is directly copied to the V2 DDFs with the following exceptions:

<table>
<thead>
<tr>
<th>System Table</th>
<th>Action on Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>X$View</td>
<td>The value of X$v$Id is copied to X$v$Sequence in pvview.ddf.</td>
<td>X$v$Sequence has auto-increment values for each row.</td>
</tr>
<tr>
<td></td>
<td>The value of X$v$Trustee, which is a new column in V2 metadata, is explicitly stored as -1.</td>
<td>The value is stored as -1 regardless of the security setting for the database. That is, if the database has security enabled or if the databases does not have security enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the V1 metadata database has security enabled, an explicit &quot;GRANT ALL TO PUBLIC&quot; is invoked for all views when the database is converted to V2 metadata.</td>
</tr>
<tr>
<td>X$Proc</td>
<td>The value of X$p$Id is copied to X$p$Sequence in pvproc.ddf.</td>
<td>X$p$Sequence has auto-increment values for each row.</td>
</tr>
<tr>
<td></td>
<td>The value of X$p$Trustee which is a new column in V2 metadata, is explicitly stored as -1.</td>
<td>The value is stored as -1 regardless of the security setting for the database. That is, the value is stored as -1 if the database has security enabled or if the databases does not have security enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the V1 metadata database has security enabled, an explicit &quot;GRANT ALL TO PUBLIC&quot; is invoked for all stored procedures when the database is converted to V2 metadata.</td>
</tr>
<tr>
<td>X$Rights</td>
<td>The value of X$r$Table is copied to X$r$Object in pvrights.ddf.</td>
<td>A default value of 1 is copied to X$r$Type in pvrights.ddf.</td>
</tr>
</tbody>
</table>

Conversion
From V2 to V1
Metadata

The current release of pvmdconv does not support conversion from V2 to V1 metadata.

Examples

For default locations of Pervasive PSQL files, see Where are the Pervasive PSQL files installed? in Getting Started With Pervasive PSQL.

Default Conversion

To convert the sample database DEMODATA to V2 metadata (assuming a default installation on a Windows platform):
pvmdconv -o 2 -d file_path\PSQL\Demodata\ -n demodata

This example results in the following:

- V2 metadata DDFs are created in the same location as the V1 metadata DDFs (file_path\PSQL\Demodata\).
- The metadata version of DEMODATA in dbnames.cfg is changed to V2 metadata.

To perform the same conversion and obtain a log of conversion diagnostic information:

pvmdconv -o 2 -d file_path\PSQL\Demodata\ -n demodata -l file_path\PSQL\Demodata\\pvmdconv_log.txt

**Convert Only DDFs**

To convert only the DDFs for the sample database DEMODATA to V2 metadata (assuming a default installation on a Windows platform):

pvmdconv -o 2 -d file_path\PSQL\Demodata\ -n demodata -DDF

This example results in the following:

- V2 metadata DDFs are created in the same location as the V1 metadata DDFs (file_path\PSQL\Demodata\).
- The metadata version of DEMODATA in dbnames.cfg is not changed.

**Change Metadata Version of Database**

To change the metadata version of the sample database DEMODATA to V2 metadata (assuming a default installation on a Windows platform):

pvmdconv -o 2 -d file_path\PSQL\Demodata\ -n demodata -v

This example results in the following:

- The metadata version of DEMODATA in dbnames.cfg is changed to V2 metadata.
- V2 metadata DDFs are not created.

Note: this example requires that the DEMODATA DDFs exist in file_path\PSQL\Demodata\ even though the DDFs are not affected by the conversion.
**Perform Conversion on a Remote Server**

To convert the sample database DEMO DATA to V2 metadata given the following:

- DEMO DATA resides on a server named “TESTSERVER”
- An administrator ID on TESTSERVER is “adminuser” and user adminuser has a password of “admin99user”
- Drive “Z” is mapped to drive “C” on TESTSERVER (assuming TESTSERVER is a Windows platform with a default installation of Pervasive PSQL).

```
pvmdconv -o 2 -d z:\file_path\PSQL\Demodata\ -n demodata -s TESTSERVER -i adminuser -c admin99user
```

This example results in the following:

- V2 metadata DDFs are created on TESTSERVER in the same location as the V1 metadata DDFs.
- The metadata version of DEMO DATA in dbnames.cfg on TESTSERVER is changed to V2 metadata.
Command Line Interface Utilities

pvnetpass

Description

pvnetpass is the Pervasive network password utility. It is a command line utility used to manage the user IDs and passwords for servers to which your client connects. When trying to connect to a server, the client looks up the server name in the registry and uses the user name and password set for that server.

If your application uses the transactional interface and connects to a Linux database engine that is configured to use BTPASSWD or PAM authentication, the application requires a set of credentials to connect to the database engine. (See Authentication in Getting Started With Pervasive PSQL.) Use pvnetpass to configure the set(s) of credentials that the application will use. Pvnetpass must be run on every machine that connects to the database engine, whether the database engine is local or remote.

If you have a global and a user entry for the same server, the user’s entry overrides the global. The user name should include the full user context. For example, in a Windows environment with domain names, specify the user as DOMAIN\user. For a Linux environment, use the user account name and the full machine DNS name. For example, mymachine.mydomain.

The pvnetpass utility can also be used by Windows clients to change their stored credentials that were saved when using the security login pop-up dialog. See also Allow Client-stored Credentials and Prompt for Client Credentials, both in Advanced Operations Guide.

Synopsis

pvnetpass [-g] [-a | -r | -m] server [-u user] [-p pwd]

pvnetpass -d

Options

- a
  Adds a server entry for a user specified by the -u parameter. If no user is specified, current user is assumed.

- d
  Displays the list of configured servers. The configured servers will display in two groups that are separated by a dashed line. The ones above the line are global entries and are only viewable by administrators who are a member of group pvs. The ones below the line are the current user’s entries. If you have a global and a user entry for the same server, the user’s entry overrides the global.
Command Line Interface Utility Reference

-g Manipulates default settings for all users. Settings created with -g can be overridden by individual users.

-m Modifies a server entry for a user specified by the -u parameter. If no user is specified, current user is assumed.

-p Specifies the password for the user. If not provided, pvnetpass prompts for a password.

-r Removes a server entry for a user specified by the -u parameter. If no user is specified, current user is assumed.

-server Server, local or remote, to which you want to add a connection entry. Server can be '*' (include the single quotes) to set the default server entry information. This default entry is used when there is no user entry for the server.

-u Specifies the name of the user. If -u is not specified, your current user name will be used.

See Also

Setting Up Client Access from a Windows Client

Examples

From current user to all servers (overrides -g)

```
pvnetpass -a '*' -p password
```

From current user to one server 'myserver' (overrides -g)

```
pvnetpass -a myserver -p password
```

From all users (-g) to one server 'myserver' using credentials joe:password

```
pvnetpass -g -a myserver -u joe -p password
```

From all users (-g) to all servers ('*'), use default credentials joe:password

```
pvnetpass -g -a '*' -u joe -p password
```

To add user ‘acctadmin’ with password ‘88sJkE5’ to the local server named ‘sles2HR’:

```
pvnetpass -a sles2HR -u acctadmin -p 88sJkE5
```

To add user ‘bholly’ with password ‘ peggysue’ to a remote server named ‘myserver’:

```
pvnetpass -a myserver -u bholly -p peggysue
```

To verify your entry was accepted, use the -d option:
Command Line Interface Utilities

```
  pvnetpass -d

  This command results in:
  
  Server: myserver
  User: bholly
  Password: (not displayed)

  To change the password with which you will connect to ‘myserver’ from your Linux client:

  pvnetpass -m myserver -u bholly -p newpassword

  To remove the entry for server ‘myserver’:

  pvnetpass -r myserver

  To add the default entry for users trying to connect to server ‘myserver’ when no user-specific entry exists:

  pvnetpass -g -a myserver -u admin -p adminpassword

  To add the default server entry in the user context (PS_HKEY_USER):

  pvnetpass -a '*' -u admin -p adminpassword

  To add the default server entry in the machine context (PS_HKEY_CONFIG):

  pvnetpass -g -a '*' -u admin -p adminpassword

  To authenticate from a Linux client to a Windows domain server (myserver) with a domain named “mydomain” and a user named “user1”:

  pvnetpass -a myserver -u mydomain\user1 -p user1password
```
rbldcli

**Description**
rbldcli is used to rebuild MicroKernel data files on your server.

**Synopsis**
rblcli [ -parameter ] file
rblcli @command-file

**Options**
For a complete discussion of the rebuild utility, see Command Line Parameters in Advanced Operations Guide.

**See Also**
Advanced Operations Guide for more information about the Rebuild utility.
Command Line Interface Utilities
Basic Troubleshooting

How to Identify and Solve Common Problems

This chapter provides information for troubleshooting and resolving the most commonly-encountered problems.

- Pervasive PSQL Message Logging
- General Troubleshooting
- Error Messages from PCC
- Frequently Asked Questions
Pervasive PSQL Message Logging

Pervasive PSQL provides various logging repositories for messages to assist you with troubleshooting. The logging falls into two broad categories:

- **All messages.** These messages include status, error, warning, and information messages. They can originate from any Pervasive PSQL component, including the license administration components.

- **Licensing messages.** These messages alert you about licensing issues and provide troubleshooting information. They originate from license administration components.

The following table summarizes the repositories.

<table>
<thead>
<tr>
<th>Repository</th>
<th>Written To By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pervasive Notification View</td>
<td>License administration components</td>
</tr>
<tr>
<td>Operating System Event Log</td>
<td>License administration components (Windows)</td>
</tr>
<tr>
<td></td>
<td>All Pervasive PSQL components (Linux)</td>
</tr>
<tr>
<td>Pervasive PSQL Event Log</td>
<td>All Pervasive PSQL components (Windows)</td>
</tr>
</tbody>
</table>

**Licensing Messages**

Several of the logging repositories emphasize licensing messages. The reason is that the Pervasive PSQL licensing components periodically verify that the key for the database engine is still valid. If a key is determined to be invalid, the key changes state from “active” to “failed validation.” The database engine functions normally for a certain number of days so that you have ample time to correct the validation failure(s).

If you do not correct the failures before the number of days ends, the key changes state again to “disabled.” The key is no longer valid and the database engine cannot access data files.

Because you need to attend to a failed validation in a timely manner, the state change of the key is brought to your attention as soon as possible. For example, a message is logged to all of the message repositories. The most evident of these is Pervasive Notification Viewer. (License Administrator also displays the state of keys, which
you can re-check at any time by initiating a validation action. See License Administration in Pervasive PSQL User's Guide.

**Change in State of Key**

The following table explains the type of message returned based on the change in state of the key.

Table 21  Message Type Based on Change in State of Key

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Change in State of Key</th>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>From</td>
<td>To</td>
</tr>
<tr>
<td>Warning</td>
<td>Active</td>
<td>Failed Validation</td>
</tr>
<tr>
<td>Error</td>
<td>Failed Validation</td>
<td>Disabled</td>
</tr>
<tr>
<td>Information</td>
<td>Failed Validation</td>
<td>Active</td>
</tr>
<tr>
<td>Information</td>
<td>Disabled</td>
<td>Active</td>
</tr>
</tbody>
</table>

Note that no messages are logged for keys with a state of “expired” (which applies only to temporary keys), or “inactive” (which applies to keys still registered on the machine from previous versions of Pervasive PSQL).
Basic Troubleshooting

Logging Frequency

The following table lists the frequency with which licensing messages are logged for particular actions.

Table 22 Logging Frequency of Messages By Initiating Action

<table>
<thead>
<tr>
<th>Initiating Action</th>
<th>Logging Frequency</th>
<th>Logging Repository¹</th>
</tr>
</thead>
</table>
| Key changes state as described in Table 21 | Immediately | • Pervasive Notification Viewer  
• Operating System Event Log  
• Pervasive PSQL Event Log |
| Key remains in failed validation state | Once a day reminder | • Pervasive Notification Viewer  
• Operating System Event Log  
• Pervasive PSQL Event Log |
| A validation action invoked programmatically through API call | Immediately | • Operating System Event Log  
• Pervasive PSQL Event Log |
| See To Display Remaining Authorizations in Pervasive PSQL User’s Guide, PvValidateLicenses() in Distributed Tuning Interface Guide, and ValidateLicenses in Distributed Tuning Objects Guide | Immediately | • Operating System Event Log  
• Pervasive PSQL Event Log |
| Warning or error messages originating from the Pervasive PSQL licensing server | Immediately | • Operating System Event Log  
• Pervasive PSQL Event Log |

¹Message logging follows a one-way hierarchy: any licensing message logged to Pervasive Notification Viewer is also logged to the Operating System Event Log and to the Pervasive PSQL Event Log. Similarly, any licensing message logged to the Operating System Event Log is also logged to the Pervasive PSQL Event Log.

Pervasive Notification Viewer

The Pervasive Notification Viewer (hereafter, just “Notification Viewer”) is an application utility for displaying messages logged by the licensing components. The purpose of the utility is to inform you of noteworthy licensing messages (see Table 22) in a noticeable but unobtrusive manner.

By default, Notification Viewer is installed with Pervasive PSQL Server, 32-bit and 64-bit, on Windows and Linux, and with Pervasive PSQL Workgroup. Also by default on Windows platforms, Notification Viewer restarts when you restart Windows.

On Windows platforms, the executable is named notifyviewer.exe. The utility provides a single running instance for a user. An attempt to start Notification Viewer when it is already running brings the GUI to the front of the application displays.
On Linux distributions, the utility is a shell script named notifyviewer. Each time the shell script executes it starts another instance of Notification Viewer. If you restart Linux, you must restart Notification Viewer. The shell script is not automatically executed when Linux is restarted.

**Command Line Options**

You can specify how you want the utility to start with the following command line options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no option)</td>
<td>If you start the utility without specifying an option, the GUI displays and a tray icon appears if the operating system supports a system tray.</td>
</tr>
<tr>
<td>-tray</td>
<td>Starts the utility with the GUI hidden and displays a tray icon. If the operating system does not support a system tray, the GUI displays (is not hidden).</td>
</tr>
</tbody>
</table>

Notification Viewer provides two interfaces: system tray icons and a graphical user interface (GUI).

**System Tray Icons Interface**

By default on Windows, Notification Viewer starts with the GUI hidden and displays its system tray icon. (On Linux, Notification Viewer starts as a GUI and displays its system tray icon if the Linux distribution supports a system tray.) After starting, the utility begins monitoring licensing messages.

If Notification Viewer detects unread messages, the tray icon visibly changes to indicate unread messages. See **Tray Icons**.

Notification Viewer also displays two types of tooltips. The mouse-over tooltip displays the number of important unread messages (if any), the total number of unread messages, or the name of the utility if all messages have been read. A balloon tooltip displays when Notification Viewer detects messages that need to be brought to your attention. On Windows, the balloon tooltip remains visible until you dismiss it directly or perform a keyboard or mouse operation. On Linux, you must click on the balloon tooltip to dismiss it.
Basic Troubleshooting

**Popup Menu**

The popup menu for the tray icon contains two menu items: “Open” which opens the GUI, and “Exit” which will terminate the utility. You right-click on the tray icon to display the popup menu.

**Tray Icons**

The following table explains the meaning of the tray icons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="" alt="Notification Viewer is running and monitoring licensing messages. This icon indicates a “normal” condition in which all messages have been read." /></td>
<td>Notification Viewer is running and monitoring licensing messages. This icon indicates a “normal” condition in which all messages have been read.</td>
</tr>
<tr>
<td><img src="" alt="Notification Viewer contains unread messages. This icon remains visible until all unread messages are read. See Left panel." /></td>
<td>Notification Viewer contains unread messages. This icon remains visible until all unread messages are read. See Left panel.</td>
</tr>
</tbody>
</table>

**Graphical User Interface**

You can open Notification Viewer GUI by double-clicking the tray icon or by right-clicking on the tray icon then clicking “Open.” By default on Linux, Notification Viewer starts as a GUI and displays its system tray icon. (If you want to change the startup behavior on Linux, pass the -tray option to the notifyviewer shell script. If the Linux distribution does not support a system tray, Notification Viewer displays the GUI but no system tray icon. In that case, start Notification Viewer by running the shell script.)

When the GUI is visible, unread messages are immediately added to the GUI. (In addition, the tray icon tooltip is shown and the icon changes to indicate unread messages.)

Pervasive PSQL tracks which records in the notification file are read or unread for each user. That is, each user displaying the GUI sees all of the messages, but whether a particular message is read or unread varies by user.

If a system tray is not supported by an operating system, the “close” command for Notification Viewer terminates the utility. If a system tray is supported, then “close” hides the GUI and the tray icon becomes the only visual representation that the utility is still running.
**Toolbar and Panels**

The Notification Viewer GUI provides a toolbar and two main panels as shown in Figure 31.

![Notification Viewer GUI](image)

**Figure 31  Notification Viewer GUI**

<table>
<thead>
<tr>
<th>GUI Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Toolbar** | Provides two options:  
  • Show only unread messages (toggle on, toggle off)  
  • Mark all messages as read |
| **Left panel** | Contains a scrollable list of messages. They are sorted with the latest messages on top and arranged in three groups: “Today,” “Previous 7 days,” and “Older.” You can expand and collapse each group.  
  Each message shows an image for the type of information, a caption string, the message date and a brief description. All text is in a bold typeface if the message is unread.  
  **To “read” a message:** click on (select) the message. Once selected, the typeface for the message changes to a regular typeface. |
| **Right panel** | Shows the message details, which provide a full description of the message, and when applicable, suggestions to resolve any issues. |

**Operating System Event Log**

The Pervasive PSQL license administration components write messages to the operating system event log on Windows platforms. All Pervasive PSQL components write messages to the operating system event log on Linux distributions.
Windows Platforms Event Logs

Windows operating systems provide a method to log events categorized as “Application,” “Security,” or “System.” Pervasive PSQL logs licensing messages only to the Windows Application Event Log. The messages are a subset of the licensing messages written to PVSW.LOG.

Any licensing message categorized as an “error” or “warning” is logged. This includes messages that result from a change in the state of a key as well as other warning and error messages (see Table 22). In addition, certain information messages are logged, such as those listed in Table 21.

Viewing Event Logs

Windows operating systems provide a graphical user interface utility called Event Viewer to view and manipulate event logs. The utility can be accessed through the Windows Control Panel or by executing the command eventvwr.msc from a command interface. Note that the Event Viewer interface for Windows XP and Windows Server 2003 differs from the one for Windows Vista and later.

Regardless which Event Viewer is used, Pervasive PSQL displays the following for an event.

- Date and Time – date and time the event occurred
- Source – Pervasive SQL
- Category – Product Keys
- Type/Level – level of the event: Information, Warning, or Error
- Event ID – 1000
- User – N/A
- Computer – name of the computer

In addition, on Windows Vista and later, the Keyword column displays “Classic” and the Log column displays “Application.” Event Viewer allows you to display additional columns, but Pervasive PSQL provides no data for them.

Linux Distributions Event Logs

On Linux distributions, all Pervasive PSQL components write messages to the standard Linux logging system, syslog. By default, syslog writes to the /var/log/messages directory. Optionally, for SQL
Connection Manager only, you can also log messages to the event.log file.

**Event.log File and Bti.ini**

Bti.ini is a Pervasive PSQL configuration file used on Linux distributions. By default, the file is located in the /usr/local/psql/etc directory.

The file lets you configure settings for the SQL Connection Manager (the “SQLManager” section in the INI file). One of the settings, LogEvent, determines the type of event messages logged to the event.log file. By default, event.log is located in the /usr/local/psql/bin directory.

<table>
<thead>
<tr>
<th>Bti.ini Parameters for SQLManager Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MgrPort</td>
<td>Sets the port number used by the SQL Connection Manager. The default is 1583.</td>
</tr>
<tr>
<td>MgrUseTransport</td>
<td>Sets the type of protocol used by the SQL Connection Manager. This must be set to TCP.</td>
</tr>
</tbody>
</table>
| LogEvent=\*msg_type\*                   | Specifies one of the following values for msg_type to indicate the type of messages logged to event.log (the default is 1):
  • 0 – no logging
  • 1 – errors only
  • 2 – errors plus warnings
  • 3 – errors plus warnings plus information messages
  • 4 – errors plus warnings plus information messages plus connect.log |
| InstallDirectory=/usr/local/psql         | Activates the connection log: /usr/local/psql/connect.log |

**Pervasive PSQL Event Log**

On Windows platforms, all Pervasive PSQL components write status, error, warning, and information messages to the Pervasive PSQL event log. (On Linux distributions, Pervasive PSQL does not use an exclusive event log. Instead, all Pervasive PSQL components write messages to the standard Linux logging system, syslog. See Linux Distributions Event Logs.)

The Pervasive PSQL event log is called PVSW.LOG and, by default, is located in the application_data_directory\PSQL\logs directory. All
Pervasive PSQL components on Windows platforms write to this log file. If two or more Pervasive-based applications are running on the same machine, they share PVSW.LOG.

**PVSW.LOG Fields**

PVSW.LOG consists of text messages in a format described in Table 23.

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Automatic date-stamp in <em>mm/dd/yyyy</em> format.</td>
</tr>
<tr>
<td>Time</td>
<td>Automatic time-stamp in <em>hh:mm:ss</em> format. Also indicates AM or PM.</td>
</tr>
<tr>
<td>Component</td>
<td>File name of component returning the error (prefix only, no extension).</td>
</tr>
<tr>
<td>Process</td>
<td>Instance ID of the component, which is the process ID of the component.</td>
</tr>
<tr>
<td>Process Name</td>
<td>Path and name of the component, truncated to the last 15 characters.</td>
</tr>
<tr>
<td>Computer Name</td>
<td>Name assigned to the machine hosting the process, truncated to the first 15 characters.</td>
</tr>
<tr>
<td>Type</td>
<td>A single character: I for Information, W for Warning, or E for Error.</td>
</tr>
<tr>
<td>Message</td>
<td>The message text which may be either a string retrieved from a resource associated with the calling component or a text string passed directly from the calling component. Some message text may contain numeric values, which may be in decimal or hexadecimal format. The characters &quot;0x&quot; precede any hexadecimal values to distinguish them from decimal values. Some message text may also contain information specific to an OEM application, such as a link to a vendor’s Web site and troubleshooting information.</td>
</tr>
</tbody>
</table>

An entry may be followed by binary data in standard hexadecimal format. There is no limit to the length of the binary data.
### PVSW.LOG Example Entry

The following shows an example of the type of data contained in PVSW.LOG.

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Component</th>
<th>Process</th>
<th>Process Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>5/10/2011</td>
<td>9:53:06 AM</td>
<td>LicenseMgr</td>
<td>9048</td>
<td>NTDBSMGR.EXE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Computer Name</th>
<th>Type Category</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>USRegion2Svr</td>
<td>W</td>
<td>License failed validation. Remaining Days: 14</td>
</tr>
</tbody>
</table>
General Troubleshooting

This section provides some basic troubleshooting procedures to help you rule out possible causes for situations you may encounter. This section covers the following topics:

- I get Error 1114 when trying to access my data
- I get an error about ServerDSN or DBQ was not found in the connection string
- I get a message about Engine components’ version is different than my client components’ version
- I can’t get to my data on the server engine
- PCC runs slowly or hangs when retrieving large record sets

I get Error 1114 when trying to access my data

or

I get an error about ServerDSN or DBQ was not found in the connection string

PCC can access remote server data sources (DSNs) using connections without client DSNs. Many desktop applications, such as Microsoft Excel and Microsoft Access, cannot do this. You must create a client DSN on your local computer to provide access to data on the server through the remote server DSN. To create a client DSN, follow the instructions in Setting Up Client Access from a Windows Client. You must first make sure that a server DSN exists on the server you want to access.

I get a message about Engine components’ version is different than my client components’ version

When a client requester first connects to an engine, the client requester compares its internal router version with the value returned from the engine by a Btrieve Version (26) call. If the client version is older than the engine, a message dialog box is displayed on the client system with the message “Engine components’ Version is different from Client’s” along with a suggestion to run Pervasive System Analyzer (PSA). The same message is also logged in the client's PVSW.LOG file.
General Troubleshooting

This message is a warning. The client is not prevented from connecting to the engine in this situation. However, Pervasive Software guarantees compatibility between engines and clients only if the clients are the same version as the database engines. When prompted by this message, if you choose not to run PSA and archive your old client components and install a newer client, you can expect the product to behave unpredictably until the client version is equal to the engine version.

Note Pervasive recommends that you use client requesters that are the same version as the database engine. If you choose, you may use a client requester that is an older version than the database engine with which it interacts. In some situations, depending on the type of SDK access method used by your application, an older version requester will not work with the database engine. Your application will be unable to communicate with the database engine. For those situations, you must use client requesters that are the same version as the database engine.

Client requesters that are a newer version than the database engine may or may not function correctly. Pervasive does not guarantee that newer versions of client requesters will function correctly with older versions of the engine. Therefore, Pervasive recommends that you avoid the use of newer version client requesters with an older engine.

I can't get to my data on the server engine

If you cannot get to data on the server engine, your most likely causes are:

- The server computer is down or the network has been interrupted
- You do not have operating system rights to access the server, or you are not logged into the correct network
- The client requester is not enabled
- The database server engine is not installed or not running
- The database server is not accepting remote connections
Basic Troubleshooting

- The remote database does not have a DSN set up to advertise on the network
- The local client does not have a DSN to access the server
- The client or server network configuration is wrong

➢ To determine the actual cause of the failure

Follow the steps below to rule out certain root causes and narrow down the possible sources of failure.

1. From the operating system on a Windows client, access the **Network** information and see if you can find the server computer to which you want to connect. If you can see the server, you can rule out that the server is down or disconnected from the network.

2. Next, try to map a drive to the file server or open a shared file on the server. If you can successfully connect to the file server and create a file on the mapped drive, then you can rule out lack of operating system rights. You can also rule out failure to login to the correct network. If you’re not logged into a particular network, you can’t access the servers on that network at all.

➢ **Note** If you are trying to create a new database on the server, to use Monitor against the remote server engine, or to configure the remote server engine, you must have administrative rights on the server, or be a member of Pervasive_Admin. A simple drive-mapping or shared-file read will not tell you whether you have administrative rights. This means you may be able to connect to the file server, but you still may not be able to connect to the database engine with Configuration, Monitor, or Create Database Wizard.

3. The next possibility is that the client requester is disabled.
   
   Start PCC then right-click the icon that represents your local client computer, then click **Properties**. Click **Access** and ensure that **Use Remote MicroKernel Engine** is selected.
   
   You can now rule out the requester as the source of the problem.

4. Next, verify that Pervasive PSQL is installed and running on the target server.
General Troubleshooting

On Windows, open **Services** under **Administrative Tools**. Verify that **Pervasive PSQL Transactional Engine** and **Pervasive PSQL Relational Engine** have been started. If not, start these services.

On Linux, type the following command at the Linux prompt on the server where the database engine is installed:

```
ps -e | egrep 'mkded'
```

If the output from the command returns at least one line containing the text “mkde,” then Pervasive PSQL is running. If you do not see this line, then you need to be logged into the root account and start the database engine by entering

```
/etc/init.d/psql start
```

You can now be certain that the server engine is installed and running.

5 The next step is to ensure that the server engine is accepting remote communication requests.

In PCC, ensure that the remote database engine is configured to accept remote requests. If you are having difficulty accessing a Windows 32-bit server engine remotely, then you must check the setting at the server itself. You must have administrative permission on the server (or membership in the Pervasive_Admin group) in order to do so. In PCC, right-click on the server, then click **Properties**. Click **Access** and ensure that **Accept Remote Request** is selected.

You can now rule out the possibility the server is not accepting remote requests.

6 Note: If your application uses pure Btrieve access only, without ODBC, then skip this step.

If everything checks out so far, but you still cannot get to the data you want to access, make sure a server DSN has been set up for your target data. Using PCC, expand the **Databases** node for that server and inspect the databases that are present. Make sure one of the databases represents the data you want to access. If so, then a server DSN has been created for your data.

If you do not find the data you want to access, but you know it is on the server, then most likely you need to set up a DSN for the given data. You must have administrative rights on the server (or be a member of the Pervasive_Admin group) to do so.
Basic Troubleshooting

Follow the instructions in Setting Up ODBC Database Access to set up a DSN for existing data files.

You can now rule out the server DSN as the source of the problem.

7 Note: If your application uses pure Btrieve access only, without ODBC, then skip this step.

If you have performed all the steps above and you still cannot get to your data, the next possibility is lack of a local client DSN for the remote data.

PCC can access remote server DSNs using connections without client DSNs. Many desktop applications, such as Microsoft Excel and Microsoft Access, cannot do this. You must create a client DSN on your local computer to provide access to the remote server DSN. To create a client DSN, follow the instructions in Setting Up Client Access from a Windows Client. You must first make sure that a server DSN exists on the server you want to access.

You can now rule out the client DSN as the source of the problem.

8 The final task to perform is to ensure that your client and server are communicating on the appropriate network protocols. By default, Pervasive PSQL ships with all network protocols enabled, so connection time may be slow as it tries all protocols, but it should eventually connect. Some application vendors disable the protocols that are not typically used by their application(s).

First, determine what protocols ought to be used on your network. If you have a Linux network or a 100% Microsoft network, then your preferred protocol is TCP/IP.

Once you know what the protocol should be, you should ensure that your server is using this protocol. You must have administrative rights on the server operating system (or be a member of Pervasive_Admin) to perform this task. In PCC, right-click on the server name then click Properties. Click Communication Protocols. Ensure that the correct protocol is listed in the Supported Protocols list and that TCP/IP is selected.
General Troubleshooting

Ensure that your client is using the same protocol. Using PCC, right-click on Local Client then click Properties. Click Communication Protocols and ensure that the correct protocol is selected in the Supported Protocols list.

9 If you have performed all of the above tasks with no success at accessing your data, refer to Pervasive PSQL Resources and Contacts for more ways to get help.

PCC runs slowly or hangs when retrieving large record sets

If this problem occurs, try increasing the amount of memory available to PCC during start up. The amount of memory you can specify is limited by the physical memory installed on your machine. You can specify a minimum and a maximum amount of memory. For example, to specify a minimum and maximum of 256 megabytes, start PCC with the following command:

```
pcc.exe -vmargs -Xms256M -Xmx256M
```

The parameter -vmargs is required if you specify the other parameters.

The parameter -Xms specifies the minimum amount of memory to allocate to PCC. The parameter -Xmx specifies the maximum amount of memory to allocate to PCC. If you specify the -Xms parameter, you must also specify the -Xmx parameter.
You may receive several different messages when attempting to create or connect to databases in PCC. This section explains the likely causes for some of the most common error messages:

- Can't retrieve database names. You don't have access rights for the operation
- Unable to connect to the specified remote server. Verify that all of the communication components are loaded on the remote server and that there are available sessions and try again
- An error was encountered while connecting to the server
- Unknown configuration properties

**Can't retrieve database names. You don't have access rights for the operation**

This error may occur when you are attempting to create a new database on the server. The most likely cause is that you are logged in as an operating system user that has neither administrative rights in the server operating system, nor membership in the Pervasive_Admin group on the server. Another likely cause is that you forgot to enter a user name and password.

**Solution:** Be sure to enter a user name and password for the remote operating system. You must have administrative rights on the server or be a member of the Pervasive_Admin group in order to create a new database on the server. [Granting Administrative Rights for the Database Engine](#) explains how to set up the Pervasive_Admin group.

For Windows 32-bit platforms, be sure that you are set up as a local user on the system, not a network user. Network users have a domain name and a back slash preceding the user name, such as BOSTON\GILBERT. Be sure that the user who is a member of the Administrators group or Pervasive_Admin group is a local user.
If you have checked permissions and your user login does in fact meet one of the criteria above, then you should also check to make sure that you are logged into the correct network. You can verify whether you are logged into the correct network by attempting to read or write to a server that you are certain uses the target operating system.

Unable to connect to the specified remote server. Verify that all of the communication components are loaded on the remote server and that there are available sessions and try again

You may receive this error when attempting to register a new remote server in PCC. There are several reasons you may receive this error:

1. You mis-typed the server name. The database client tried to connect to a server that does not exist.
   
   **Solution:** Double-check the name of the server, and make sure you can see it in your Network Neighborhood, spelled exactly how you entered it.

   If you know the server exists but you can't see it in your Network Neighborhood, make sure that you are logged into the correct network. Ask your network administrator for help.

2. The server user count has expired. If you have been using a temporary license, you will get this message for connection attempts after the license has expired.

   **Solution:** Run License Administrator to check the status of licenses installed on the server. In the window that appears, you can see detailed status information on each license that has been applied to your server. If your license has expired, purchase a permanent license from your reseller or from Pervasive Software.

3. There are no available sessions on the server. If you have a heavy load of users on the server, or if you have configured the server with a small number of sessions, you may receive this error.
Basic Troubleshooting

Solution: Run Monitor to check the usage of sessions available on the server. In Monitor, select Options then Connect and connect to the server in question. Then choose MicroKernel then Communications. In the window that appears, find Total Remote Sessions. If the Peak value and the Maximum value are the same, then it is likely that you have run out of sessions.

4 The remote database server is not running.
Solution: Make sure that the remote database engine is running, or ask your network administrator to do so.

5 The remote database server is not accepting client requests.
Solution: Set the properties to ensure that the remote database engine is configured to accept remote requests. You must have administrative permission on the server (or membership in the Pervasive_Admin group) in order to do so. In PCC, right-click on the server name in Pervasive PSQL Explorer, then click Properties. Click Access and ensure that the Accept Remote Request option is selected.

An error was encountered while connecting to the server

The most likely cause of this error is using the wrong operating system user name or password in an attempt to connect to the server.

Other possible causes include:

- The operating system may be expecting the user to change his/her password on the first logon. This situation occurs if, in the User Manager, you have selected the User Must Change Password at Next Logon checkbox.
- If the user is a member of another group with lesser permissions, the lesser permissions will override the greater permissions. A user always has the most restrictive permissions of any group to which the user belongs.

Solution: Double-check the spelling of the user name and the password. Make sure the user and password have been set up on the remote server operating system.

Inspect the user’s account information on the server. Make sure the operating system is not expecting the user’s password to be changed at the next logon. Make sure the user is not also a member of a group that has restricted permissions.
For Windows 32-bit platforms, be sure that the user is set up as a local user on the system, not a network user. Network users have a domain name and a back slash preceding the user name, such as BOSTON\GILBERT. Be sure that the user who is a member of the Administrators group or Pervasive_Admin group is a local user.

Unknown configuration properties
It is possible, but unlikely, that PCC may retrieve configuration properties from the database engine that are invalid. Please contact Pervasive Customer Support to report such error conditions. See Pervasive PSQL Resources and Contacts.
Frequently Asked Questions

This section answers some of the questions that customers ask most frequently. A list of the questions is provided below:

Installation
- Will I lose my data files if I uninstall my existing version of the product, or install a new version?
- Why do I not see in PCC Pervasive PSQL Explorer the “plug-in” product that I just installed or upgraded?
- What type of client install should I do?
- How can I be sure what service pack level of client I am running?
- Is Pervasive PSQL supported on a Terminal Server?
- Can I install Pervasive PSQL in a Failover environment? or Can I install Pervasive PSQL in a Clustering environment?
- Can I install Pervasive PSQL in a Load Balancing environment?
- Can I install Pervasive PSQL on a server running Btrieve v6.x or earlier?
- How do I keep my Workgroup Engine from starting up automatically when I reboot?

PCC
- How do I start PCC on Linux?

Security
- When do I login using an operating system user and password, and when do I login using a database user and password?
- Why do I get a “log in failed” message when I have a Pervasive_Admin group defined or I have administrator rights?

User Count, Session Count, Data In Use
- How do I authorize a User Count Upgrade?
- How does the Workgroup engine keep track of how many people are accessing the data? If people access the data with two engines at the same time, what happens?
Frequently Asked Questions

- Does the Workgroup engine use concurrent or per-seat licensing?
- I am thinking of moving from PSQL Server to PSQL Vx Server, which means moving from licensing based on user count to licensing based on data usage. How do I determine the appropriate size for my Vx Server license?

Networking

- How do I know which protocol I am using for communication? I can see other systems in Network Neighborhood but I can't get to my data.

Difficulty Accessing Data

- I upgraded from Btrieve v6.x or earlier to Pervasive PSQL v11 SP3. Now I get error messages telling me that a file is inaccessible when everybody else can get to it. What's wrong?
- I have files sitting on the server that are shared and yet Pervasive PSQL cannot read them. What's wrong?
- I am using SQL queries to create a definition for an old table. The resulting record size is off. Why?
- I want to convert my data file version from 9 back to file format version 8, 7, or 6. How do I do this?

ODBC and DDFs

- How can I tell if I can use ODBC to access my data files?
- How can a hard-coded filepath in a DDF be changed?
- What is the best way to ensure that my data dictionaries (DDFs) are safe?
- How can I tell whether I have non-standard DDFs?
- Can I mix and match DDFs from different databases?
- What happened to DDF Sniffer?
- I have two similar Btrieve files, and I created a DDF for the first one. Since they are similar, can I use the same DDF on the second Btrieve file?
- I have owner names set on my Btrieve files. After I created a DSN, I cannot open the files using ODBC. What's wrong?
- Is there a client side requester for the SRDE?
Basic Troubleshooting

- Is ODBC the only method of access for Pervasive PSQL?
- Is there a single database file housing all the data, data definitions, stored procedures, security, table relationships, and so on as in some other products?
- Does the SQL engine (SRDE) have scheduler capabilities to run stored procedures or other types of scripts designed to access and affect data?

Upgrading from Btrieve 6.15

- Is there a tool that replaces Xtrieve?
- Upgrading and Migration

Upgrading and Migration

- Where can I find information on migration from earlier product versions to Pervasive PSQL v11 SP3? Where can I find migration and compatibility information?
- When I create a table using an existing Btrieve file, the wizard displays fewer columns than there are in the Btrieve file. What's wrong?

DEMODATA Sample Database

How do I restore DEMODATA to its installation defaults?

1. Start PCC if it is not already running. (See Starting PCC on Windows.)
2. From the PCC Main menu, click File ➤ New SQL Document or click . (See To start SQL Editor for a new SQL query.)

The Select Database dialog appears.
3 Click **DEMODATA** in the list then click **OK**.
SQL Editor appears as a new tab view in PCC.

4 Click **File** ▶ **Open**.

5 Navigate to the location of the **DEMODATA.sql** file then click **Open**.

For default locations of Pervasive PSQL files, see **Where are the Pervasive PSQL files installed?** in **Getting Started With Pervasive PSQL**.

6 Click **SQL** ▶ **Execute All SQL Statements**, press **F10**, or click .

**DEMODATA.sql** deletes the existing tables then re-creates them using the installation defaults. The restored tables are empty (no data).

Any new tables that you have created as part of **DEMODATA** are not affected.
Basic Troubleshooting

7 Use the Bulk Data Utility (BDU) to populate the tables with data (see bdu).

   a. Open a command prompt.
   b. Go to the "restore" directory for DEMODATA. An "SDF" file exists for each table in DEMODATA.
   c. For each SDF file, use BDU to load the data into the table. For example, to load the data for the "Billing" table use the following command. The comma is the field delimiter in the file.

        bdu demodata billing billing.sdf -t ,

   Note The "-t," parameter is required.

   The BDU utility returns how many rows of data were loaded.

Miscellaneous

- I dumped Btrieve records to a file and now I can't read the file. What happened?
- How do I run Pervasive PSQL in trace mode?
- Does garbage collection occur in the data files and indexes? For example, is space from deleted records recovered or reused?
- Is database shadowing available, allowing a complete up-to-date second copy of the database to exist on another drive or machine?
- What is the mechanism that allows the database to be backed up online? What happens if the server goes down in the middle of a backup with many open transactions?
**Installation**

Frequently asked questions about installation.

**Will I lose my data files if I uninstall my existing version of the product, or install a new version?**

When you uninstall Pervasive PSQL or install a new version of Pervasive PSQL, your data files and DDFs are never affected. Even when Pervasive System Analyzer archives Pervasive PSQL files, or even if you have your data files in the same directory as Pervasive PSQL files, your data files are not affected.

**Why do I not see in PCC Pervasive PSQL Explorer the “plug-in” product that I just installed or upgraded?**

See Situations Requiring That You Clear PCC Cache.

**What type of client install should I do?**

If you are not sure, always select complete. This option performs a standard installation, which makes it easier to troubleshoot if problems occur.

**How can I be sure what service pack level of client I am running?**

If you are using Pervasive PSQL v11 SP3, start Monitor or Maintenance, choose Help then About from the menu, and check the Build Level.

**Is Pervasive PSQL supported on a Terminal Server?**

Support for both the Server and Workgroup engines on Terminal Server has been available since Pervasive.SQL 2000i, SP 4. Pervasive.SQL 2000i, SP 3 provided support for the Server engine. Pervasive.SQL 2000i, SP 2 provided supports only for the client software.

**Can I install Pervasive PSQL in a Failover environment? or Can I install Pervasive PSQL in a Clustering environment?**

Yes. Pervasive PSQL is cluster compatible with Microsoft Failover Clustering, Microsoft Cluster Service, and Linux Heartbeat. See the chapter High Availability Support in Advanced Operations Guide.
Basic Troubleshooting

Can I install Pervasive PSQL in a Load Balancing environment?
That is not supported at this time.

Can I install Pervasive PSQL on a server running Btrieve v6.x or earlier?
No, you cannot run Pervasive PSQL and Btrieve 6.x on the same computer at the same time.

How do I keep my Workgroup Engine from starting up automatically when I reboot?
You must remove it from the Windows Startup group. Check the operating system documentation for Startup.

PCC
frequently asked questions about PCC.

How do I start PCC on Linux?
Certain requirements must be met before you can start PCC on Linux. See Starting PCC On Linux.

Security
frequently asked questions about security.

When do I login using an operating system user and password, and when do I login using a database user and password?
This may seem confusing at first, but in fact there is only one rule: use a database login only after you have already successfully connected to the server and are attempting to access a database directly. Up until this point, you should use an operating system login.

For example, if you run Monitor or Configuration to work with a remote server engine, you are prompted for a password. In both cases, you must supply a user name and password for an operating system account that has administrative permissions on the remote system, or an account that is a member of Pervasive_Admin. This applies also when you are creating a new database.

Once you start to work with the data itself, then you must supply a database user and password, if prompted. If database security is turned off, then you would never need a database user name or password. In this case, you would only need an operating system user
Frequently Asked Questions

and password to perform administrative tasks, as noted in the preceding paragraph.

Why do I get a “log in failed” message when I have a Pervasive_Admin group defined or I have administrator rights?

The settings for the Pervasive PSQL services can affect whether or not you have permission to log in to the machine where the database engine is running. The settings apply whether or not you use a Pervasive_Admin group. If you change the Log on as setting for a Pervasive PSQL service to This account, you must change the user rights policy Act as part of the operating system for the account. Otherwise, remote log in fails.

For example, the Monitoring utility requires that you log in to the operating system on the machine where the database engine is running. You will receive a message that log in failed if the account specified for This account cannot act as part of the operating system.

Note that even the Administrator account requires that you set the user rights policy for Act as part of the operating system.

See Services Settings and Log In Authority for a complete discussion and the steps to change the user rights policy.

User Counts

frequently asked questions about user counts.

How do I authorize a User Count Upgrade?

Refer to the tasks discussed in the chapter License Administration.

How does the Workgroup engine keep track of how many people are accessing the data? If people access the data with two engines at the same time, what happens?

The Workgroup engine keeps track of users just as the Server engine does. Each license specifies a user count. A user count allows the specified number of computers to connect to the Pervasive database engine concurrently. Pervasive PSQL identifies machines by serial number, so all connections with the same serial number are recognized as coming from the same machine. A machine with multiple NICs, for example, is recognized as the same machine.

Each workstation that accesses Pervasive PSQL as a client session counts as one user. Multiple applications on a single client computer
are counted as one user, not separate users. Each Terminal Server session also counts as one user.

Pervasive PSQL uses one user count for each unique incoming protocol from the same client computer session. If one application uses TCP/IP and another application uses SPX, two licenses are counted if both applications run on the same machine. If different address formats of the same protocol are used, only one user is counted. For example, if one application uses IPv4 and another uses IPv6, only one user is counted if both applications run on the same machine. IPv4 and IPv6 are just different address formats of the TCP/IP.

The Pervasive PSQL Workgroup engine includes a trial license. The license for the Workgroup engine cannot be deauthorized.

Only one engine is ever permitted to access data files at a time. The second engine to try to open the files gets locked out, because the engines open the data files in exclusive mode (non-file sharing) so that corruption cannot occur.

**Does the Workgroup engine use concurrent or per-seat licensing?**

Concurrent. Refer to User Count License Model.

I am thinking of moving from PSQL Server to PSQL Vx Server, which means moving from licensing based on user count to licensing based on data usage. How do I determine the appropriate size for my Vx Server license?

Capacity Usage Viewer was designed for this purpose. Because it records both peak daily session use and peak daily data usage, you can see how much of each you use over time. You can use this information to make a reasonable determination of what license size you need for Vx Server. Refer to Capacity Usage Viewer for information about how to access and use Capacity Usage Viewer.
Networking

**How do I know which protocol I am using for communication?**

*I can see other systems in Network Neighborhood but I can’t get to my data.*

Start Pervasive System Analyzer and click **Next** on the Welcome screen. On the following screen, check the box **Test Network Communications** and make sure all the other boxes are not checked. Click **Next**. In the following screen, cancel the selected protocols that you do not want to test. Click **Browse** to select the drive that you have mapped to the installation directory on the server. You must have a mapped drive; UNC names are not supported. Click **Next** to run the network tests. The results window tells you if there are any significant problems with your networking.

Difficulty Accessing Data

Frequently asked questions about accessing data.

**I upgraded from Btrieve v6.x or earlier to Pervasive PSQL v11 SP3. Now I get error messages telling me that a file is inaccessible when everybody else can get to it. What’s wrong?**

Use Pervasive System Analyzer to be sure that all components from previous versions of Btrieve or Pervasive PSQL have been archived. Then, make sure your configuration settings are correct. Find the file pvsw.log and check for error messages indicating a status code 8505 or 8517. These status codes indicate that attempts were made to use a local Workgroup engine to read the data files. (Note that hexadecimal values in message strings written to pvsw.log are preceded by “0x” to distinguish them from decimal values.)

Start PCC (see **Starting PCC on Windows** in Pervasive PSQL User’s Guide). Right-click on **Local Client** then click **Properties**. Click **Access**. Ensure that the option **Use Local MicroKernel Engine** is not check marked and that **Use Remote MicroKernel Engine** is check marked.

**I have files sitting on the server that are shared and yet Pervasive PSQL cannot read them. What’s wrong?**

How are the files shared? Pervasive does not support mapping a drive letter using Redirected mapping under Microsoft, or using the hidden Admin share (C$) under Windows 32-bit platforms.
Basic Troubleshooting

Make sure that users have appropriate operating system login credentials to access the file server.

Run Pervasive System Analyzer and choose the Network Communications Test to be sure that you have proper connectivity.

_I am using SQL queries to create a definition for an old table. The resulting record size is off. Why?_

Starting with Pervasive.SQL 2000, fields that allow null values have an additional byte defined at the start of the field. This byte is the null indicator byte. You can work around this in one of two ways:

If you are using SQL statements to create a new table definition, enter the statement SET TRUE NULLCREATE=OFF. For the remainder of your current session, any tables that you create will use the old record structure without the extra byte for each nullable column.

If you do not wish to use SQL statements, you can get the field sizes to align properly by creating all columns as not nullable.

_I want to convert my data file version from 9 back to file format version 8, 7, or 6. How do I do this?_

If the files you wish to convert are serviced by a remote Server or Workgroup engine, you must have Administrator permissions on the remote system in order to perform these tasks. You must also have a network drive mapped to the remote data files.

In PCC, right-click the server name where the data files are located then click Properties. Click Compatibility and set Create File Version to the file version to which you want to convert. Click OK. Click Yes to restart the engines. These changes result in new files created to be in the version selected.

Run the Maintenance utility then click Options then File Information Editor. Click Load Information and choose the data file that you want to convert. Click Create and specify the name of the new, empty data file you want to create with the older version format. Click OK to create the file. Close the File Information Editor window, but do not exit Btrieve Maintenance Utility.

From the menu, select Data then Copy. Enter the name of the source data file and the name of the target data file (your newly created file with the older version file format). Click Execute to copy the records into the older version file. After the copying has finished, if you need the new data file to have the same name as it did previously, save your
original data file with a different name, then save your new file using the original file name.

**ODBC and DDFs**

Frequently asked questions about ODBC and dictionary files.

**How can I tell if I can use ODBC to access my data files?**

There are several ways to find out. First, look for .DDF files where the data files are located. If you see them, then most likely you can access the database using ODBC. Because it is possible to have DDF files located in a different directory, you should also use PCC to determine whether a database has been created for the data files you want to access. Finally, you can ask your application vendor whether their application uses ODBC to access the data files.

**How can a hard-coded filepath in a DDF be changed?**

In PCC, right click on the database to which the table belongs, then click **Properties**. Click **Directories**. Change the value for **Dictionary Location**.

It may appear that the path has not changed. To confirm the change, open the X$File system table and look at the Xf$Loc field for the given user table. If you cannot see the system tables in PCC, expand **System Objects**.

You can also use the ALTER TABLE USING statement in SQL to change the data file used by a particular table. Refer to SQL Engine Reference for further information.

**What is the best way to ensure that my data dictionaries (DDFs) are safe?**

Always keep a backup copy of your DDFs. Anytime you make changes to the runtime DDFs, be sure to make a backup copy of the DDFs before making changes. If you are turning on database security for the first time, you should make a backup copy of the dictionaries without security, and a backup copy with security.

**How can I tell whether I have non-standard DDFs?**

If you can edit your DDFs with a Btrieve utility, it means that you do not have standard dictionary files. A standard dictionary file does not permit direct Btrieve access. This lock out is a safety feature that ensures only the SRDE can write to the dictionary. DDFs are very
special files that must remain synchronized with each other and with the data files at all times.

Standard dictionaries do not have case sensitive table names or field names. That is, the column definitions for column Xf$Name in file.ddf and column Xe$Name in field.ddf have the Case flag set, meaning the values are case insensitive.

DDFs are Btrieve files and thus can be opened and viewed (not updated) using the Function Executor. This is one way to confirm the contents of file.ddf or field.ddf.

On some non-standard dictionaries, the DDFs file.ddf, field.ddf, and/or index.ddf do not exist. Such dictionaries do not work with our products. For example, if you see a file called x$file.ddf, instead of file.ddf, you can assume your DDFs are non-standard.

Non-standard DDFs are unlikely to work properly with Pervasive PSQL Control Center or the relational engine.

Can I mix and match DDFs from different databases?

A complete set of DDF files must be considered a unit. No DDF file from one database may be intermixed with DDFs from a different database.

What happened to DDF Sniffer?

DDF Sniffer was added to the Pervasive product line with the acquisition of Smithware in 1998. It is no longer available as a separate product. Its functionality has since been largely replaced by the Pervasive PSQL Control Center and DDF Builder.

I have two similar Btrieve files, and I created a DDF for the first one. Since they are similar, can I use the same DDF on the second Btrieve file?

The answer depends on how similar the files are. If the two files differ only in the number of records, you can use the same DDF file. If there are any differences at all in the number, order, names, or types of fields or indexes, you cannot use the same DDF. In other words, you can only use the same DDF if the record structure of the two files is identical.
Frequently Asked Questions

I have owner names set on my Btrieve files. After I created a DSN, I cannot open the files using ODBC. What's wrong?

If Btrieve files have owner names on them, you must use database security for ODBC access. Turn on database security in PCC. See To turn on security using Pervasive PSQL Explorer and Owner Names and Security in Advanced Operations Guide.

⚠️ Caution Do not forget the Master user password. You cannot turn off security or perform administrative tasks within the database without it. You may want to make a backup copy of your DDFs before turning security on, in case you forget the password.

Next, you must grant the Master user access to the data files that have owner names defined. You can grant the access by issuing this SQL statement for each table that has an owner name:

```
GRANT ALL ON my_table 'ownername' TO Master
```

When you enter the statement, substitute the actual name of your table and the appropriate owner name for that table, as indicated above. Remember that each data file corresponds to an ODBC table. If you don't know which table corresponds to which data file, see use PCC to find out. Right-click on the table in PCC then click Properties. Click Information in the tree. The FileName field shows you the file that is referenced by that table definition.

If security is important, then you must create users and assign permissions for all users expected to access the database. You do this by using the CREATE GROUP and GRANT statements in SQL. You can also add users and groups with PCC. See Users and Groups in Advanced Operations Guide.

If security is not important to you, you can avoid creating many users and assigning privileges by granting access to PUBLIC, which means anyone on your network can access the data. You can use this statement:

```
GRANT ALL ON my_table 'ownername' TO PUBLIC
```

Is there a client side requester for the SRDE?

There is no DOS requester support for SQL applications, but the Pervasive PSQL client software for Windows includes ODBC client...
Basic Troubleshooting

components allowing you to connect to a remote SRDE server engine.

**Is ODBC the only method of access for Pervasive PSQL?**

Definitely not! In addition to ODBC and the time-tested Btrieve API, you can also develop applications using our OLE DB provider, our JDBC driver, our pure Java interface, or our ActiveX controls.

**Is there a single database file housing all the data, data definitions, stored procedures, security, table relationships, and so on as in some other products?**

No. Pervasive PSQL stores data in separate files, one file per relational table definition. The meta data, such as data definitions, user/group definitions, and so on, are stored in a set of DDF files, where each file ends in the extension “.ddf.”

**Does the SQL engine (SRDE) have scheduler capabilities to run stored procedures or other types of scripts designed to access and affect data?**

The SRDE does not have a scheduler.

**Upgrading from Btrieve 6.15**

Frequently asked questions about Btrieve 6.15.

**Is there a tool that replaces Xtrieve?**

There is no direct replacement, but you should consider using Crystal Reports for Btrieve as an excellent upgrade from Xtrieve for reporting on and querying Btrieve data.
Frequently Asked Questions

**Upgrading and Migration**

Frequently asked questions about upgrading and migration.

*When I create a table using an existing Btrieve file, the wizard displays fewer columns than there are in the Btrieve file. What's wrong?*

Btrieve files contain a limited amount of information about the structure of the file. The table creation wizard can figure out some field definitions using the indexes, but after the indexes are exhausted, data segments may remain that contain more than one actual field. The wizard has no way of interpreting the contents. You must use your detailed knowledge of the record structure to split out these fields and build a table definition that matches all the fields in the record.

The procedure for this task is provided in Advanced Operations Guide.

*Where can I find information on migration from earlier product versions to Pervasive PSQL v11 SP3? Where can I find migration and compatibility information?*

Getting Started With Pervasive PSQL contains an entire chapter that provides detailed instructions on how to upgrade.

If your application uses Scalable SQL or ODBC, then you should review the Application Migration Guide available on the web site.

**Miscellaneous**

Frequently asked questions about miscellaneous topics.

*I dumped Btrieve records to a file and now I can't read the file. What happened?*

If you use the Btrieve Maintenance Utility to save/dump the records, the resulting file contains the binary image of each record. Unless the record consists entirely of character data, it may not be readable to the human eye.

The only way that Pervasive PSQL can dump a record in ASCII readable format, is by reading the DDFs to get a description of the total contents of the record. Btrieve only has the record length, the data type of indexes and length of the indexes. Btrieve does not have information on how to interpret the entire contents of the record.
How do I run Pervasive PSQL in trace mode?

Server

You must have administrator privileges on the machine where the engine is located that you want to run in debug mode.

1. Using PCC, right-click on the desired Server engine then click Properties.

2. Click Debugging and set the value for Trace Operation to On.

3. Click OK.

You do not need to restart the engine.

See also Trace Operation in Advanced Operations Guide.

Note After tracing operations, you should turn off Trace Operation, making sure to click Edit ➤ Apply when finished. You will notice slower performance if you run Pervasive PSQL in trace mode.

Windows Client

Run the PSA network connectivity tests to verify network connectivity. See Test Active Installation Tasks. Also refer to the Knowledge Base, available at the Pervasive Software website, for information about particular issues.

In addition, client tracing is available for troubleshooting certain types of low-level problems. Generally, low-level tracing is not required, so this type of tracing is intended for use by trained support staff. Your product vendor or Pervasive Software Support will explain how to conduct low-level client tracing.

Does garbage collection occur in the data files and indexes? For example, is space from deleted records recovered or reused?

Yes, space from deleted records is re-used on subsequent inserts. Space in files is never de-allocated back to disk. If index balancing is turned on, unused space in index pages is also re-used. See Rebuild Utility Concepts in Advanced Operations Guide.
Is database shadowing available, allowing a complete up-to-date second copy of the database to exist on another drive or machine?

Pervasive PSQL does not contain specific functionality for this, but many customers have successfully used hardware mirrored drive arrays and solutions like Vinca's (now acquired by Legato) Standby Server to provide this functionality. Pervasive.SQL 2000i SP3 and later supports Pervasive's data replication product, Pervasive DataExchange.

What is the mechanism that allows the database to be backed up online? What happens if the server goes down in the middle of a backup with many open transactions?

Continuous Operations allows you to put a set of data files in a special “safe mode” so that they can be safely backed up while in use. While data files are in Continuous Operations mode, they are not modified, and special delta files store the results of any database operations. After the backup is complete, the data files must be removed from Continuous Operations mode, at which time the changes stored in the delta files are rolled into the live files.

If the server goes down while files are in continuous operations mode, the next time the data file is accessed, the database engine detects the existing delta file and rolls in the changes at that time.

You can put data files into Continuous Operations mode by using the BUTIL -STARTBU command or Maintenance utility described in Advanced Operations Guide.
Basic Troubleshooting
Pervasive PSQL Resources and Contacts

A Guide to Pervasive PSQL Customer Information Resources

Pervasive Software strives to ensure that your experience with Pervasive PSQL is successful. This chapter describes the resources and information available to you as a valued customer of Pervasive Software.

The following variety of resources can help you get answers to your questions, troubleshoot problems, and interact with the Pervasive team as well as with other customers.

Pervasive Software strives to ensure that your product installation is easy and successful. If you encounter problems during or after the installation that are not covered in the user documentation, please contact Pervasive Software and we will address your problem promptly.

The following table lists a variety of resources to help you get answers to your questions, troubleshoot problems, and interact with the Pervasive team as well as with other customers.
### Thirty-Day Free Technical Support

Your purchase of Pervasive products entitles you to 30 days of free technical support for installation and configuration problems. The free technical support is limited to two support issues (referred to as “tickets”) during the 30 days.

### Table 24 Pervasive Software Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
<th>Contact Information</th>
</tr>
</thead>
</table>
| Pervasive PSQL Web site         | The site is a great source for everything Pervasive PSQL, such as the following:  
                              | • Product downloads for Pervasive PSQL, Pervasive AuditMaster,                | http://www.pervasivedb.com          |
|                                 | • Pervasive Backup Agent, and Pervasive DataExchange                        |                                     |
|                                 | • Technical support and Knowledge Base                                      |                                     |
|                                 | • Discussion forums                                                         |                                     |
|                                 | • Software development kit (SDK) downloads                                  |                                     |
|                                 | • Product documentation, white papers, and technical papers                 |                                     |
|                                 | • Component downloads such as tools, solutions, and code samples             |                                     |
|                                 | • Company contacts, and more!                                               |                                     |
| Pervasive PSQL FTP Site         | An FTP site is available to upload files that you want to provide Technical Support | ftp://ftpsupport.pervasive.com      |
| Pervasive PSQL Newsgroup        | The Pervasive PSQL newsgroup is managed by the end-user community, posting and answering questions as they wish.  
                              | Note that the discussion forums on the Pervasive PSQL Web site have largely replaced activity on the newsgroup. | news://comp.databases.btrieve      |
| Pervasive PSQL Printed          | Printed versions of each manual are available for purchase separately, or you may purchase the entire documentation set. | E-mail database@pervasive.com       |
| Documentation                   |                                                                             | or telephone 1 800 287 4383.        |
## Index

### A
- Access conflict 327
- Access methods 334
- Access rights
  - cannot perform the operation 316
  - problems due to lack of O/S rights 312
- Access to database
  - setting up 33
- ACS
  - viewing 101
- Active Directory
  - create Pervasive_Admin group on domain controller 29
- Admin share not supported 329
- Administrative rights
  - for Active Directory 27
  - granting 26
  - granting for "this account" 28
  - granting on Linux 31
  - granting on Windows 27
  - troubleshooting 327
  - user rights policy 29
- Administrator
  - login 32
- Administrator user
  - fast user switching 53
- Authorization
  - configuring proxy server 129
  - telephone 158
- Autostart
  - keeping Workgroup engine from autostarting 326
- Available sessions
  - error in PCC if none available 317

### B
- Backup
  - online 337
- Bcfg utility 249
- Bdu utility 250
- Bmon utility 257
- Bound database
  - properties for database 93
- Btadmin utility 258
- Bti.ini 307
  - event log 307
- Btrieve
  - and terminology 11
  - owner names 333
  - testing transactional interface 239
- Bulk Data Utility 250
- Bulk Data utility
  - constraints 253
  - error logging 253
  - examples 254
- BUTIL
  - Linux and 260

### C
- Cache
  - clearing 60
- Capacity-based license model 127
- Case comparisons
  - in Table Editor 167
- Cell
  - defined 8
- Changing file path in DDF 331
- Clear a key 155
- Client
  - definition of 4
  - DSN 33
  - DSN setup 39, 41
  - installation 325
  - password utility for use with 294
  - setting up database access 39
  - testing connectivity to database engine 238
  - troubleshooting network protocols 314
- Client DSN
  - on Linux workstation 41
  - set up with ODBC Administrator 39
- Clilcadm utility 262
- Clustering 326
Collate
  in Table Editor 168
Collating sequence
  setting for column in Table Editor 181
Column
  defined 7
Column name
  in Table Editor 167
Columns
  allowing index duplicates in Table Editor 195
  arranging order of index segment in Table Editor 194
  deleting in Table Editor 176
  deleting index in Table Editor 190
  deleting index segment in Table Editor 193
  inserting in Table Editor 175
  inserting index segment in Table Editor 191
  modifying index in Table Editor 190
  modifying index segment in Table Editor 192
  naming in Table Editor 176
  selecting in Table Editor 176
  setting collating sequence in Table Editor 181
  setting data type in Table Editor 177
  setting default in Table Editor 182
  setting for nulls in Table Editor 180
  setting index in Table Editor 184
  setting precision in Table Editor 179
  setting primary key in Table Editor 183
  setting scale in Table Editor 179
  setting size in Table Editor 178
  specifying index as modifiable in Table Editor 196
  specifying sort order for index in Table Editor 195
  viewing those defined for a table 103
Columns page
  in Table Editor 163
  interface in Table Editor 166
Command-line
  utility to execute SQL statements from 287
  utility to rebuild data files from 297
Communication
  testing network connectivity 238
Configuration
  bti.ini 37
database code page 91
defined 123
engine DSN 37
object properties in PCC 123
PCC connection encoding 92
properties for database 90
properties for database engine 71
properties for services 69
properties for table 101
server O/S rights required 312
setting general preferences 51, 64
setting general preferences for PCC 215
setting preferences for Data Grid 65
setting preferences for external tools 67
setting preferences for Monitor 65
setting preferences for PCC Grid 64, 65
setting preferences for PCC SQL Editor 65
setting preferences for PCC Text window 65
setting preferences for PCC window views 65
setting preferences for Table Editor 65
setting preferences in PCC for external tools 67
Conflict, access 327
Connection strings
  'ServerDSN' or 'DBQ' not found in 310
Continuous Operations
  behavior after server failure 337
Continuous operations
  restriction for files with same name 172
Control Center. See PCC
Convert data files to older file format 330
Create database 98
Create Database Wizard
  creating a Client DSN 39
  creating an Engine DSN on Windows 36
CREATE GROUP 333
CREATE USER 333
D
Data
  files where stored 334
  importing 104
  location of files for database 93
Data compression
  viewing 101
Data definitions
  where stored 334
Data directories for database 93
Data files
convert to older file format 330
not affected by install/uninstall 325
rebuilding with utility 297
re-use of space 336
sharing of DDFs 332
Data Grid
setting preferences 65
Data in use
determine total limit in License Administrator
GUI 152
Data replication 337
Data source name
engine DSN 36
Data table 101
Data types
Columns page in Table Editor 163
Database
access 33, 36
access for client workstation 39
as object in PCC 73, 90
bound 93
cannot retrieve names error 316
code page properties 91
concepts 2
creating 98
data directories 93
defined 2, 8
dictionary location 92
error 7039 when creating 98
importing data 250
integrity enforced 94
logging out from in PCC 90
mirroring 337
PCC connection encoding property 92
security 111
security vs OS security 326
set up access for Workstation or Workgroup
engine 36
set up access on Linux 37
set up access on Windows 36
setting properties for 90
setting up on client 39
starting engine 20
stopping engine 20
structures 7
structures and terms 7
test ODBC connection 274, 278
testing client connectivity to 238
Database encoding 91
Database engine
as object in PCC 70
java.lang.UnsatisfiedLinkError error on Linux 59
logging in to in PCC 71
logging out from in PCC 70
reconnecting to in PCC 71
registering in PCC 70
running as service in PCC 68
services dependent on 20
setting services properties 69
setting startup policy 69
start and stop on Windows 21, 22
starting and stopping on Linux 25
starting or stopping 68
starting server 20
stopping 20
SWT no more handles error on Linux 59
unable to connect to error on Linux 60
Database Management System
defined 3
functions of 3
Database security
logging in as an administrator 32
Database Workgroup Engine
start and stop on Windows 23
DataExchange
displaying in PCC after installation 60
displaying in PCC after upgrading 60
Dbmaint utility 37, 264
DBMS. See Database Management System
DBQ not found error 310
DDF location for database 92
DDFs
changing file path 331
creating for old data 330
detecting non-standard 331
for ODBC access 36
keeping safe 331
mixing among databases 332
path viewing 101
sharing among data files 332
Deauthorization telephone 160
Default object in Table Editor 168
Definitions
client 4
requester 4
Delete
DSN 51
Deleted records, re-use of disk space 336
Delimited data
loading into a database 250
DEMODATA
recovering default tables 322
restoring to installation defaults 322
Dependency
of services of database engine 20
Detect non-standard DDFs 331
Determining
service pack level 325
Dictionary
location of files for database 92
Dictionary path
error 7039 when creating a database 98
viewing 101
Disk space
re-use of 336
Displaying
key information 154
remaining authorizations 154
Documentation 14
Online help 16
DSN
client 33, 39
client DSN needed for many applications 314
client DSN on Linux client 41
concepts 33
engine 33, 36
engine DSN on a Linux server 37
File DSN not supported 33, 40
recreating 35, 51
removing 51
test connection to database 274, 278
Dsnadd utility 41, 267
Dump file
cannot read 335
Duplicate pointers
viewing number of 101

E
Encoding
database code page 91
Engine
as object in PCC 70
DSN 33
DSN setup 36
logging in to in PCC 71
logging out from in PCC 70
message that components differ from client
components 310
reconnecting to in PCC 71
registering in PCC 70
running as service in PCC 68
setting properties for 71
setting services properties 69
setting startup policy 69
starting or stopping 68
stopping Workgroup under fast user switching 54
testing client connectivity to 238
tray icon with fast user switching 54
Error 1114 310
Error 7039 when creating a database 98
Error was encountered message 318
Event log
and bti.ini 307
events logged for Linux 307
message logging 300
PVSW.LOG 307
Events
scheduling 334
Expired licenses
hiding 149
Export
data from table 105
Export files
cannot read file 335
Export table schema 106
External tools
adding to PCC 67
F
Failover 325
FAQ. See Frequently asked questions
Fast user switching 53
running Workgroup engine 53
tray icon with 54
types of users for 53
Field
defined 7
Field.ddf 331
File
viewing file name associated with a table 103
viewing location of table 101
File DSN not supported 33, 40
File format
converting data to older 330
File path
changing in DDF 331
File system security 17, 45
File version
viewing 101
File.ddf 331
Files with same name
restriction for 172
Find non-standard DDFs 331
Finding
service pack level 325
Foreign key
adding in Table Editor 197
deleting in Table Editor 199
modifying in Table Editor 199
Foreign keys page
in Table Editor 164
Freespace threshold
viewing 101
Frequently asked questions 320

G
Garbage collection
on disk 336
GRANT 333
Granting
administrative rights 26
administrative rights for “this account” 28
administrative rights on Linux 31
administrative rights on Windows 27
Grid
adding rows of data 222
changing data in 221
copying data from 225
deleting rows of data 223
in PCC 63
refreshing data in 224
setting preferences 64, 65
using scalar functions 224
view in SQL Editor 204
Group
as object in PCC 111
assign user to 119

H
How
to apply a user count upgrade 327

I
Identifying
service pack level 325
Import Data Wizard
restrictions 104
Importing
data 104
data into a database 250
data into table 104
IN DICTIONARY
clause in exporting table schema 106
Increasing
user count license 134
Increasing user count licenses 327
Index
defined 8
Index balancing
viewing 101
Index.ddf 331
Indexes
allowing duplicates in Table Editor 195
arranging order of segment in Table Editor 194
creating unique 186
deleting in Table Editor 190
deleting segment in Table Editor 193
inserting segment in Table Editor 191
modifying in Table Editor 190
modifying segment in Table Editor 192
part 188
setting in Table Editor 184
specifying as modifiable in Table Editor 196
specifying sort order in Table Editor 195
viewing number of 101
viewing those defined for a table 103
Indexes page
  in Table Editor 164
Installation
  client FAQ 325
data files not affected by 325
testing with PSA 230
Integrity enforced
properties for database 94
Interfaces
  supported for programming 334
Isql utility 274, 278

J
Java.lang.UnsatisfiedLinkError error on Linux 59
Join
  defined 10

K
Key
  system 101
  validation 154
Key only file
  viewing 101
Key state
  message logging 301
Keys
  displaying information 154
Keys. See License key

L
Legacy data
  creating DDFs for 330
Legacy file format
  convert data files to 330
License Administrator 125
  applying keys 148
  applying user count increases 148
  CLI syntax 141
  CLI tasks 154
command line interface 141
concepts 126
deauthorization concepts 126
deleting a key 149
determine session count limit 151
determine total data in use 152
determining total user count 151
displaying applied license 150
displaying help 153
graphical user interface 137
GUI tasks 146
GUI Visual Reference 137
hide expired licenses 149
increasing user count 134
license key 126
licenses for prior releases 137
obtaining user count 127
select a computer for license administration 147
start as stand-alone application 146
start from PCC 146
tasks with 145
user count 126
License Administrator (CLI)
  apply a key offline 157
  apply a key remotely 156
  applying keys 155
  deleting a key 155
  deleting a key offline 158
  deleting a key remotely 157
  deleting a key without deauthorizing 155
  displaying all applied licenses 154
  displaying help 156
  displaying information about keys 154
  displaying remaining authorizations 154
  performing a validation 154
  validating a key 154
License authorization
  configuring proxy server 129
License count
  increasing 327
License count. See User count
License key
  administering licenses with License Administrator 125
  and user count 126
  applying a key 148, 155
applying a key offline 157
applying a key remotely 156
applying a user count increase 148
deleting a key 149, 155
deleting a key offline 158
deleting a key remotely 157
deleting a key without deauthorizing 155
determining total data in use 152
determining total user count 151
displaying all applied licenses 150, 154
for prior releases 137
obtaining user count 127
platforms 135
session count limit 151
starting License Administrator 146
License Key Platform 135
License model
capacity-based 127
License models 126
Licensing
workgroup engine 328
Limited user
fast user switching 53
Linux
btadmin utility 258
butil utility 260
client DSNs 41
clicadm utility 262
dbmaint utilities 37
dbmaint utility 264
directory owner required to create a database 98
dsnadd utility 41, 267
generateDSN utility 37
errors trying to start PCC 58
naming databases 37
psql force command 25
psql start command 25
psql stop command 25
starting and stopping database engine 25
SWT no more handles engine error with PCC 59
troubleshoot starting PCC 58
unable to connect to database engine error on PCC 60
utilities 248
utility to change passwords 285
utility to register components in Pervasive registry 284
Linux utility to edit Registry 282
Load balancing 326
Load data
bulk data utility 250
Loading
database engine 20
Local
defined 9
Log
name of PSA log file 232
specifying name for PSA log file 241
Log file 235
Log on authority for service 28
Log on failed
troubleshooting 327
Log out
from database 90
LogEvent
Bti.ini parameter 307
Logging messages 300
Login
OS vs DB 326
M
Map Root not supported 329
Memory
increasing amount available to PCC 315
viewing modules loaded in 236
Message logging 300
change in state of key 301
frequency 302
operating system event log 305
PVSW LOG 306, 307
repositories 300
Metadata
files where stored 334
utility to convert 289
Metadata conversion utility 289
Mgrport
Bti.ini parameter 307
MgrUseTransport
Bti.ini parameter 307
MicroKernel Database Engine
defined 3
Microsoft Access
accessing Pervasive PSQL data using 47

Microsoft Excel
accessing Pervasive PSQL data using 45

Mirroring of database 337
MIKDE. See MicroKernel Database Engine
Monitor
O/S rights required 312
setting preferences 65
Monitoring
bmon utility 257
multiple processors
use of 336

Naming databases
dbmaint utility 37
from a Linux server 37

Network
determining protocol 329
protocol troubleshooting 314
setting protocol for connectivity test 239
testing connectivity with PSA 238
verifying communications 329
Network password utility 294
Network protocol
determine the one in use 329
Non-standard DDFs
detecting 331
Notification Viewer 302
command line options 303
graphical user interface 304
system tray icons 303
Null
in Table Editor 167
Nulls
and Columns page in Table Editor 163
setting for column in Table Editor 180
Number of duplicate pointers
viewing 101
Number of indexes
viewing 101
Number of unused pages
viewing 101
Number of records
viewing 101

Object properties
accessing in PCC 63

ODBC
accessing data 45
concepts 33
determining if used 331
test utility 274, 278
ODBC standard 33
Offline applying of key 157
Offline deleting of key 158
Online backup 337
Online help 16
Operating system
security vs DB security 326
troubleshooting access rights 312
Outline window
in PCC 64
Outline window view
in SQL Editor 207
running all SQL statements 220
sizing 226
Owner names
and SQL security 333

Page size
viewing 101
Pages
viewing number of unused 101
Partial indexes 188
Password
troubleshooting wrong 318
utility to change on Linux 285
utility used to manage 294
Passwords
utility used to manage 294
PCC
accessing object properties 63
additional utilities 66
clean parameters 60
clearing cache 60
configuration 123
database engine object 70
database engine running as a service 68
database object 73, 90
displaying a plug-in after plug-in installation 60
displaying DataExchange and other plug-ins 60
textual characteristics 61
etditors and window view within 61
external services 67
genral preferences 64, 215
Grid 63
group object 111
increasing memory available to 315
java.lang.UnsatisfiedLinkError error on Linux 59
logging in to server engine 71
logging out from database 90
logging out from server engine 70
Outline window 64
overview
Pervasive PSQL Explorer 62
preferences 51
preferences for external tools 67
preferences for window views 65
reconnecting to a server engine 71
registering a remote server engine 70
running slowly or hanging 315
security 111
services 68
setting connection encoding 92
setting database code page 91
setting database engine properties 71
setting database properties 90
setting preferences for external tools 67
setting table properties 101
specifying amount of memory to use 315
SQL Editor 63
stored procedure object 110
SWT no more handles error on Linux 59
Table Editor 64
table object 101
Text window 64
trigger object 110
troubleshooting starting on Linux 58
unable to connect to database engine error on Linux 60
user object 111
user-defined function object 110
view object 110
vmarg parameter to increase memory 315
window view characteristics 61
window views 65
PCC connection encoding 92
PCC hangs 315
PCC hangs, See PCC
PCC running slowly 315
Performing
validate 154
Pervasive Control Center hangs, See PCC
Pervasive Notification Viewer
Pervasive Notification Viewer See Notification Viewer
Pervasive PSQL
client, defined 34
explained 2
need to recreate DSNs 35
Server 12
server, defined 34
Workgroup 12
Pervasive PSQL Control Center. See PCC
Pervasive PSQL Event Log 307
Pervasive PSQL event log
field descriptions 308
Pervasive PSQL Explorer
accessing object properties 63
expanding tree objects 62
within PCC 62
Pervasive PSQL utilities. See Utilities
Pervasive Registry
utility to edit on Linux 282
Pervasive registry
utility to register components 284
Pervasive Software
how to contact 339
Pervasive System Analyzer. See PSA
Pervasive Admin security group 26
use with Active Directory 29
Platform for license key 135
Precision
GUI object in Table Editor 167
Preferences
database default for current PCC session 215
external tools 67
for Data Grid 65
for Monitor 65
for PCC 64
for PCC DSN entries 51
for PCC SQL Editor 65
for PCC Table Editor 65
for PCC Text window 65
setting for Grid 65
setting for PCC Grid 64
setting in PCC for external tools 67
window views in PCC 65
Primary key
setting for column in Table Editor 183
Programming interfaces supported 334
Properties
accessing in PCC 63
bound database 93
data directories location for database 93
dictionary location for database 92
integrity enforced for database 94
setting database code page 91
setting for database 90
setting for database engine 71
setting for services 69
setting for table 101
setting PCC connection encoding 92
Protocol
determining network 329
setting for PSA connectivity test 239
Proxy server
configuring for authorization 129
PSA
log file name 232
name of wizard executable 236
setting protocols for network connectivity test 239
specifying name for log file 241
starting 236
test active installation 230
testing Btrieve application 239
testing network connectivity 238
testing relational interface 240
testing SQL application 240
testing transactional interface 239
view modules 230
viewing Pervasive modules 236
PSA, See Pervasive System Analyzer
Psc utility 279
return codes 280
Psql force command 25
Psql start command 25
Psql stop command 25
Psregedit utility 282
Psregsvr 284
Psdbpass utility 285
Psddi utility 287
Psmdconv utility 289
Psnetpass utility 294
authenticating to a domain server 296
PVSW.LOG 307
client and server compatibility 311
R
Rbldcli utility 297
Rebuild
data files with utility 297
Reconnect
to database engine in PCC 71
Record
defined 8
length 101, 330
viewing variable 101
Recover
default data for DEMODATA sample database 322
Recreating DSNs 35, 51
Redirected mapping not supported 329
Referential constraints
listing 90
Register
database engine in PCC 70
Registry
utility to register components on Linux 284
Registry utility on Linux 282
Relational
testing interface with PSA 240
Remaining authorizations
displaying 154
Remote
defined 9
Remote applying of key 156
Remote database access 39
Remote deletion of key 157
Removing
DSN 51
Replication 337
Requester

defined of 4
for SQL Relational Database Engine 333
troubleshooting client 312

Restore

DEMODATA sample database to installation defaults 322

Restrictions

Import Data Wizard 104

Rights

administrative 26
granting administrative for "this account" 28
granting administrative on Linux 31
granting administrative on Windows 27
if changed from system account 28

Row

defined 8

S

Sample database

recreating DEMODATA 322

Scalar functions

DATE in SQL Editor 224
TIME in SQL Editor 224
TIMESTAMP in SQL Editor 224
using within SQL Editor Grid 224

Scale

GUI object in Table Editor 167

Scheduling events 334

Schema

defined 9
exporting table 106

Security 111
administrative rights 26
as object property in PCC 111
file system 17, 45
logging in as administrator 28, 32
OS login vs DB login 326
owner names vs SQL security 333
tasks for 112

Separator

selecting statement separator 217

Server

troubleshooting network protocols 314
ServerDSN not found error 310

Service

database engine running as 68
log in authority for "this account" 28
setting properties 69
setting startup policy for database engine 69
starting or stopping database engine 68
stopping Workgroup engine when using fast user switching 54

Service pack

identifying level 325

Services

psc utility for 279
psc utility return codes 280

Session count
determine limit in License Administrator GUI 151

Set up database access 33

Shadowing of database 337

Shared files, troubleshooting 329

Size

in Table Editor 167

SM P. See Multiple processors

SQL

CREATE GROUP 333
CREATE USER 333
GRANT 333
testing relational interface 240
utility to execute statements 287

SQL Editor 63

adding rows of data within Grid 222
changing data within Grid 221
clearing results in Text window 225
copying data from Grid 225
creating SQL query or script 216
creating stored procedures 226
creating triggers 226
creating user-defined functions 226
creating views 226
deleting rows of data within Grid 223
deleting stored procedures 227
deleting triggers 227
deleting user-defined functions 227
deleting views 227
enabling SQL execution commands 219
finding text 218
Grid 204
identifying setting 215
modifying stored procedures 227
modifying triggers 227
modifying user-defined functions 227
modifying views 227
opening SQL script 216
Outline window view 207
overview 202
redoing an action 218
refreshing data in Grid 224
running all SQL statement 220
running all SQL statement in Outline window view 220
running single SQL statement 219
selecting SQL statement separator 217
selecting text 219
selecting text in Text window 225
setting database context for query 214
setting preferences 65
sizing Outline window view 226
SQL statement separators 203
starting by showing all table records 214
starting for new query 213
stored procedures 208
Text window view 205
triggers 208
typing comments in 218
undoing an action 218
used in SQL View of Table Editor 210
user-defined functions 208
using scalar functions within Grid 224
views 208
SQL Relational Database Engine
defined 4
SQL requester 333
SQL script
creating in SQL Editor 216
opening in SQL Editor 216
SQL statements
comments in 218
copying in Table Editor 199
creating query or script in SQL Editor 216
enabling commands to execute 219
finding text in 218
opening SQL script in SQL Editor 216
redoing typing in SQL Editor 218
running all in SQL Editor 220
running in SQL Editor Outline window view 220
running single in SQL Editor 219
selecting text in 219
separators in SQL Editor 203
setting database context in SQL Editor for query 214
starting SQL Editor 213
starting SQL Editor by showing all table records 214
typing comments in SQL Editor 218
undoing typing in SQL Editor 218
SQL View
in Table Editor 210
SQL view
in Table Editor 164
Start database engine on Linux 25
Start up
clearing Workgroup engine from autostarting 326
Starting
database engine 20
Startup policy
clearing database engine 69
Statement separator
clearing 217
troubleshooting 217
Stop database engine on Linux 25
Stopping
database engine 20
Workgroup engine under fast user switching 54
Stored procedure
as object in PCC 110
creating in SQL Editor 226
deleting in SQL Editor 227
in SQL Editor 208
modifying in SQL Editor 227
Structured Query Language
utility to execute statements 287
Structures
database 7
SWT no more handles error on Linux 59
Symmetric Multiprocessing. See Multiple processors
System account rights 28
System data key
viewing 101
System key
  viewing 101
System tables 101

T
Table
  as object in PCC 101
  creating data for 104
data 101
defined 8
export data 105
exporting schema 106
import data 104
system 101
  viewing alternate collating sequence 101
  viewing data compression 101
  viewing dictionary path 101
  viewing file version 101
  viewing freespace threshold 101
  viewing index balancing 101
  viewing key only file 101
  viewing location of file 101
  viewing name 101
  viewing number of duplicate pointers 101
  viewing number of indexes 101
  viewing number of records 101
  viewing number of unused pages 101
  viewing page size 101
  viewing physical file name associated with 103
  viewing record length 101
  viewing system data key 101
  viewing variable record blank truncation 101
  viewing variable records 101
Table columns
  viewing 103
Table Editor
  adding a foreign key 197
  allowing index duplicates 195
  arranging order of index segment 194
columns page 163
columns page interface 166
  copying SQL statement 199
  creating an index 184
deleting a column 176
deleting a foreign key 199
deleting an index 190
deleting an index segment 193
foreign keys page 164
  identifying tables with changes pending 173
in PCC 64
indexes page 164
  inserting an index segment 191
  inserting columns 175
  modifying a foreign key 199
  modifying an index 190
  modifying an index segment 192
  naming a column 176
overview 162
  saving changes for edited table 174
  saving changes to edited table 174
selecting columns 176
setting a column data type 177
setting column case 180
setting column collating sequence 181
setting column default 182
setting column nulls 180
setting column precision 179
setting column primary key 183
setting column scale 179
setting column size 178
setting preferences 65
  specifying index as modifiable 196
  specifying sort order for index 195
SQL View 210
SQL view 164
  starting for a new table 171
  starting for an existing table 172
tasks performed with 169
  undoing or redoing changes 174
viewing SQL statements applicable to the table 173
  viewing table data 173
work area pages 162
working with columns 172
working with foreign keys 172
working with indexes 172
Table file location
  viewing 101
Table indexes
  viewing 103
Table name
  viewing 101
viewing physical file name associated with 103
Table schema
  exporting 106
Tables
  exporting schema 106
Telephone authorization 158
Telephone deauthorization 160
Terminal Server 325
Terminal Server Licensing 128
Terminology
  Btrieve usage 11
  client 4
  requester 4
Test active installation 234
Test network 234
Test relational engine 235
Test transactional engine 234
Text window
  in PCC 64
  setting preferences 65
Text window view
  clearing results in 225
  in SQL Editor 205
  selecting text in 225
Tools
  adding external to PCC 67
Trace
  run in trace mode 336
Trace mode
  running in 336
Tracing 336
Transactional
  testing interface with PSA 239
Tray icon
  with fast user switching 54
Trigger
  as object in PCC 110
  creating in SQL Editor 226
  deleting in SQL Editor 227
  in SQL Editor 208
  modifying in SQL Editor 227
Troubleshooting
  client DSN not available 314
  disabled client requester 312
  network outage 312
  no server (engine) DSN available 313
  OS permissions problems 312
  PCC running slowly or hanging 315
  server not accepting requests 313
  server not running 312
  SQL Statement Separator 217
  wrong network protocol settings 314
Type
  in Table Editor 167

U
Unable to connect error
troubleshooting 317
Unable to connect to database engine error on Linux 60
Unauthorized access
to data file with owner name 333
Uninstalling
data files not affected 325
Unloading
database engine 20
Unused pages
  viewing number of 101
Upgrading 335
  user count 327
User
  as object in PCC 111
User count 126
  administering with License Administrator 125
  determining total 151
  how to apply 327
  increasing 134
  license key 126
  obtaining 127
  troubleshooting expired license 317
  workgroup engine 328
User count license model 126
User name
  OS vs DB 326
  troubleshooting wrong 318
  utility used to manage 294
User rights policy
  act as part of the operating system 29
  for "this account" 29
User switching 53
User-defined function
  as object in PCC 110
creating in SQL Editor 226
deleting in SQL Editor 227
in SQL Editor 208
modifying in SQL Editor 227

Users
assign to group 119
types of for fast user switching 53

Uses alternate collating sequence
viewing 101

USING
clause in exporting table schema 106

Utilities
btadmin 258
butil 260
clicadm 262
dbmaint 37, 264
dsnadd 41, 267
for Linux 248
License Administrator 125
overview 1, 19, 299
psc 279
w64clicadm 262

V
V1 metadata
conversion utility 289
V2 metadata
conversion utility 289
Value
defined 7

Variable record blank truncation
viewing 101

Variable records
viewing 101

Version
new, data files not affected by installing 325
old, data files not affected by uninstalling 325
viewing file 101

View
as object in PCC 110
creating in SQL Editor 226
deleting in SQL Editor 227
in SQL Editor 208
modifying in SQL Editor 227
Pervasive modules loaded in memory 236
View loaded Pervasive modules 234

View modules 230
Vmargs
PCC parameter to increase memory 315

W
W64clicadm utility 262
Windows
fast user switching 53
running Workgroup engine with fast user
switching 53
start and stop Database Workgroup Engine 23
types of users for fast user switching 53
Windows services
psc utility 279
psc utility return codes 280
Windows XP operating system
tray icon with fast user switching 54
Wizard
PSA 236
Workgroup engine
database access 39
keeping from autostarting 326
licensing 328
simultaneous access 327
stopping under fast user switching 54

X
Xtrieve
replacement for 334