## What is SQL Central?

SQL Central is a Windows tool for administering Sybase SQL Anywhere databases. It lets you, as the database administrator, manage objects in your database (tables, procedures, and so on), as well as helping you perform common database-wide tasks (creating databases, backing them up, and so on).

SQL Central makes these tasks simpler by providing you with an easy-to-use, familiar interface much like the Windows 95 Explorer.

To accomplish a task, you simply point and click. For example, to delete a table, you select the table in the **Tables** folder and click **Delete** in a menu.

SQL Central helps you with more complicated tasks (such as unloading a database) by providing utility wizards which walk you through the correct procedure step by step.

SQL Central lets you use the type of interface that you’re most comfortable with - drop-down menus, pop-up menus, toolbars, keyboard shortcuts, or drag-and-drop.

---

**Notes:**

- SQL Central can also administer Watcom SQL 4.0 databases running on a SQL Anywhere 5 server, and can upgrade databases (Watcom SQL 3.2 or 4.0, or SQL Anywhere 5) to the latest SQL Anywhere 5 database format.

---

Related Topics

## Requirements for SQL Central

SQL Central requires the following:

- an Intel-based PC running Windows 95 or Windows NT 3.51 (or later)
- SQL Anywhere 5.0 (or later)
- a SQL Anywhere 5 database or a Watcom SQL 3.2/4.0 database, with a user account having DBA authority

---

Related Topics

## Starting SQL Central

### Under Windows 95 or Windows NT 4.0 (or later):

- In the **Start** menu, click **Programs > Sybase SQL Anywhere 5.0 > SQL Central**

### Under Windows NT 3.51:

- In Program Manager, double-click the **SQL Central** icon in the **Sybase SQL Anywhere 5.0** group window.

---

Related Topics

## How to get help

There are several kinds of help available for SQL Central:

- How-to help for accomplishing tasks in SQL Central (which you are reading now)
- Tooltips and status bar messages that appear when you point at a toolbar button or menu item
- Step-by-step tutorials on using SQL Central, in the SQL Anywhere User's Guide
- Sybase customer support

### To get how-to help for SQL Central, do one of the following:

- In SQL Central, click **Help Topics** in the **Help** menu, or press **F1**.
- In Program Manager (or the **Start** menu), open **SQL Central Help**.

---

### Notes:

- This help file describes how to use SQL Central (a part of the Sybase SQL Anywhere database product). For complete help on SQL Anywhere, see the **SQL Anywhere User's Guide**.

---

Related Topics

## Contacting Sybase Customer Support

If you have problems, questions, or suggestions for Sybase SQL Anywhere, please contact Sybase Customer Support in your part of the world:

- North America and International (excluding Europe)
- Europe

---

Related Topics

## Customer Support - North America/International

**Sybase, Inc.**, 6475 Christie Avenue Emeryville, California USA 94608

### General Inquiries

- Tel: 617-564-7353
- FAX: 303-294-3739

### Sales

(upgrades, other product orders, and reseller inquiries/orders)

- Tel: 800-8-SYBASE
- Annual support sales: 800-395-3525
  (toll-free in North America)

### Technical Support

- Installation and registration support
  - North America 519-884-0702
  - Latin America 713-977-0752
  - Singapore 65-378-0140
- Annual support 800-937-7693
- Pay-per-issue support 508-287-1950
- Customer service fax 508-369-4992
- BBS: 508-287-1850
- CompuServe: type GO SYBASE
  or GO POWERSOFT
  or GO WATCOM
- Microsoft Network: type GO SYBASE
  or GO POWERSOFT
  or GO WATCOM
- Internet FTP site: ftp.sybase.com
  ftp.powersoft.com
  ftp.watcom.com
- World Wide Web: www.sybase.com
  www.powersoft.com
  www.watcom.com

SQL Anywhere customer support services are subject to then-current price, terms, and conditions.

### Faxline

Technical support and product information is available 24 hours a day using the Faxline fax-back system.

To reach Faxline, call 508-287-1600.

---

Related Topics

## Customer Support - Europe

**Sybase Europe**, Windsor Court, Kingsmead Business Park, High Wycombe, Bucks., HP11 1JU, UNITED KINGDOM

### Sales

(upgrades, other product orders, and reseller inquiries/orders)

| | |
|---|---|
| ◆ Austria | 0660 6714 |
| ◆ Belgium | 0800 1 5562 |
| ◆ Denmark | 80 01 02 07 |
| ◆ France | 05 90 81 35 |
| ◆ Germany | 0130 81 88 62 |
| ◆ Italy | 167 872036 |
| ◆ Netherlands | 060 222102 |
| ◆ Norway | 800 11012 |
| ◆ Spain | 900 99 4417 |
| ◆ Sweden | 020 791 705 |
| ◆ Switzerland | 155 4881 |
| ◆ United Kingdom | 0800 44 44 55 |
| ◆ All other locations | +44 1494 555 599 (UK) |
| Fax | +44 1494 555 595 (UK) |

### Technical Support

| | |
|---|---|
| ◆ Belgium | +32 2 352 3333 |
| BBS | +32 2 352 3303 |
| ◆ Czech Republic | +42 2 311 4596 |
| ◆ France | +33 1 41 02 34 56 |
| ◆ Germany | +49 6122 9232 32 |
| ◆ Netherlands | +31 20 651 8402 |
| ◆ Norway | +47 51 52 12 20 |
| ◆ Spain | +34 1 593 26 36 |
| ◆ United Kingdom | +44 1494 555566 |
| BBS: | +44 1494 555533 |
| ◆ CompuServe: | type GO SYBASE<br>or GO POWERSOFT<br>or GO WATCOM |
| ◆ Microsoft Network: | type GO SYBASE |
| ◆ | or GO POWERSOFT<br>or GO WATCOM |
| ◆ Internet FTP site: | ftp.sybase.com<br>ftp.powersoft.com<br>ftp.watcom.com |
| ◆ World Wide Web: | www.sybase.com<br>www.powersoft.com<br>www.watcom.com |

For information on technical support in countries not listed, please call the appropriate sales line.

SQL Anywhere customer support services are subject to then-current price, terms, and conditions.

### Faxline

Technical support and product information is available 24 hours a day using the Faxline fax-back system.

To reach Faxline, call +44 1 494 555522. In Belgium, call +32 2 352 3305.

---

Related Topics

## Connecting to a database

For most database operations in SQL Central, you must be connected to the database.

How you connect to a database depends on whether it is already running (and if it is, whether the server that it’s running on is visible in SQL Central).

### Click one of the following:

- The database is already running on a server that I can see in SQL Central
- The database isn't running, or is running on a server that I can't see

Once you connect to a database, it is shown in the left panel of the main window, under the server that it is running on. The user that you connected as is shown in brackets after the database name.

You can then administer the database by navigating and selecting objects that belong to the database.

You can make subsequent connections to a given database easier and faster by using a connection profile.

### Notes:

- SQL Central is a tool intended for database administrators. If you use SQL Central to connect to a database using an account that does not have DBA authority, most SQL Central features will be unavailable to you.
- SQL Central does **not** show all servers running on your machine or on the network. It only shows those that are running the databases that you are connected to in SQL Central.
- To start a database without connecting to it, see Starting a database without connecting .

Related Topics

## Connecting to a running database on a visible server

### To connect to a running database on a server shown in SQL Central:

- Open the desired server.
  {button ,AL(`server open',0,`',`')} How?
- Do one of the following:
  - Click the database (shown as a disconnected database).
  - Click **Open** in the **File** menu.
  - Right-click the database, then click **Open** in the pop-up menu.
- In the connection dialog, enter your User ID and Password.
- Click **OK** to connect to the database.

---

Related Topics

## Using the connection dialog

### To connect to a database on a server that isn't shown in SQL Central, do one of the following:

- Click **Connect** in the **Tools** menu.
- Click the **Connect** icon in the main toolbar.

  Either of these steps opens the connection dialog. How you use the connection dialog depends on the state of the database that you want to connect to. The cases include:

  - Connecting to a lone running database on a lone stand-alone engine
  - Connecting to a running database if you can't see the server
  - Connecting to a database not currently running

---

**Notes:**

- To connect to a running database on a server shown in SQL Central, see Connecting to a running database on a visible server
- You can make subsequent connections to a given database easier and faster by using a connection profile.

---

Related Topics

## Connecting to a lone running database on a lone stand-alone engine

**To connect to a database that is the only one running on a single stand-alone engine on your PC:**

- Open the database connection dialog.
- Enter your User ID and Password.
- Click **OK** to connect to the database.

---

Related Topics

## Connecting to a running database if you can't see the server

### To connect to a running database if you can’t see the server in SQL Central:

- Open the database connection dialog.
- Enter your User ID and Password.
- Click **More** to expand the dialog.
- If you are running more than one stand-alone engine, or if you're using a SQL Anywhere client to connect to a network server, enter a Server Name.
- Enter the Database Name.
- Click **OK** to connect to the database.

---

Related Topics

## Connecting to a database not currently running

### To connect to a database that is not currently running:

- Open the database connection dialog.
- Enter your User ID and Password.
- Click **More** to expand the dialog.
- Enter a Server Name.  
  {button ,PI(`',`IDH_DBX_server_name_details')} More details
- Enter a Database Name.  
  {button ,PI(`',`IDH_DBX_database_name_details')} More details
- In the **Database Startup** section, enter the path and name of a Database File.  
  You can click **Browse** to locate the file using a standard dialog.
- If you're using a stand-alone engine, click the **Local** option.  
  If you're using a SQL Anywhere client to connect to a network server, click the **Network** option.
- To start the database with specific command-line parameters, click the **Custom** option, then click the **Custom** button to open the command-line-parameter dialog. Enter the command and parameters and click **OK** to exit the dialog.
- Click **OK** to start the database and connect to it.

---

Related Topics

## Specifying a server name

If you enter a name for the server, the database will try to use a running stand-alone engine or SQL Anywhere client of that name (depending on the **Local** or **Network** options).
If no engine/client of that name is running, one will be started automatically using that name.

If you leave the **Server Name** blank for an engine or client, the database will use the first active engine that it finds on the local machine (regardless on the **Local** or **Network** options).
If no engines are running, one will be started automatically, and will be named the same as the database name (or the root of the database file, if the database name is blank). For example, starting a database file of **SADEMO.DB** yields a server named **SADEMO**.

## Overview of connection profiles

When you first connect to a database, you typically enter a user name, password, and various optional connection parameters (database name, server name, and so on).

To make subsequent connections to that database easier and faster, you can create a connection profile that saves that group of parameters under a name that you supply. From then on, you can use that profile to connect to the database in one step.

To use and manage connection profiles, click **Connection Profiles** in the **Tools** menu. This opens the connection profile dialog, where you can:

- create a new connection profile
- connect to a database using a profile
- edit an existing profile
- set a profile to connect automatically when SQL Central is started
- delete profiles

---

Related Topics

## Creating a new connection profile

### To create a new connection profile for a database:

- Open the connection profile dialog.
- Click **New**.
- In the profile name dialog:
  - Enter a unique name for the profile.
  - Choose **Sybase SQL Anywhere** as the type.
  - Click **OK** to exit the dialog.
- In the connection dialog, enter the connection parameters.
- Click **OK** to create the connection profile.

You can then:

- use it to connect to a database
- make SQL Central automatically connect to it on startup.

---

Related Topics

# Connecting to a database using a profile

## To connect to a database using an existing connection profile:

- Open the connection profile dialog.
- Do one of the following:
  - Select the profile in the list, then click **Connect**.
  - Double-click the profile.

Once you connect to a database, it is shown in the left panel of the main window, under the server that it is running on. The user that you connected as is shown in brackets after the database name.

You can then administer the database by navigating and selecting objects that belong to the database.

---

Related Topics

## Editing a connection profile

### To edit the parameters of an existing connection profile:

- Open the connection profile dialog.
- Select the profile in the list, then click **Edit**.
- In the connection dialog, make your changes to the connection parameters.
- Click **OK** to save your changes to the profile.

---

Related Topics

## Setting profiles to connect at startup

### To make a profile connect automatically when you next start SQL Central:

- Open the connection profile dialog.
- Select the profile in the list.
- Click **Set Startup** to toggle the current setting.

There is no restriction on the number of automatic connections that you can have.

---

Related Topics

## Deleting connection profiles

### To delete one or more connection profiles:

- Open the connection profile dialog.
- Select the profiles in the list.
- Click **Delete**.

---

Related Topics

## Layout of the main window

The main SQL Central window is very similar to the Windows 95 Explorer.

The main window is split into two panels:

- The **left panel** shows the object tree (a hierarchical view of SQL Anywhere servers and databases). Note that the left panel only shows the containers in the object tree. It does not show objects that are not containers of other objects. For example, in the **Users & Groups** folder, the left panel shows groups (which are containers), but not users (which are shown in the right panel instead).
- The **right panel** shows the contents of the currently selected container. You can change the appearance of the right-panel view to show large icons, small icons, a multi-column list, or a detailed list.

The main window also has a toolbar and status bar.

Once you connect to a database, you can administer it by navigating and selecting its objects in the main window.

---

Related Topics

## Navigating in panels

In the left and right panels, there are several ways to navigate through the object tree.

To open a container, double-click it. This expands the container so that:

- the left panel shows all containers in it, and
- the right panel shows its contents.

To expand a container in the left panel without changing the right panel, click its ⊞ button in the left panel.
To close an open container, double-click it (or click its ⊟ button in the left panel). This collapses the container in the left panel by hiding the containers in it.

To move up one level in the object tree (for example, from a given table to the **Tables** folder), click the **Up One Level** icon in the main toolbar.

You can also move up in the object tree by choosing objects in the object-tree drop-down combo box (above the left panel). This is handy when the part of the tree that you want to select has scrolled out of sight in the left panel.

You can also use the arrow keys to navigate through the object tree.
{button ,PI(`',`IDH_DBX_navigating_keyboard_details')}   How?

---

Related Topics

## Navigating using the keyboard

You can use the arrow keys to navigate through the object tree:

- In the left panel:
  - The up/down keys move up and down in the list of containers.
  - The right/left keys open and close the selected container.
- In the right panel:
  - The up/down keys move up and down in the list of objects.
  - The right/left keys scroll the panel.

## Selecting objects in panels

In either panel, clicking an object selects that object.

If you select a container in the left panel, the right panel shows its contents.

In the left panel, you can only select one object at a time. In the right panel, you can select several objects at a time.

The **Select All** command in the **Edit** menu selects all objects in the right panel.

The **Invert Selection** command reverses which objects are selected and which are not.

Right-clicking an object opens a pop-up menu of commands for that object.

---

Related Topics

## Changing the right-panel view

In the right panel of the main window, you can show objects as:

- **Large Icons** - shown left to right, top to bottom, labeled underneath
- **Small Icons** - shown left to right, top to bottom, labeled on right
- **List** - small icons, but shown top to bottom, left to right (as many columns as necessary)
- **Details** - small icons, but shown top to bottom, with columns for name, comment, and other properties

### To change the view, do one of the following:

- Choose the desired view from the **View** menu.
- In an empty spot in the right panel, right-click and choose the desired view in the pop-up menu.
- Click the appropriate view icon in the main toolbar.

---

Related Topics

## Refreshing the main window

While you are working in SQL Central, other users can make changes to the database’s structure (provided they have sufficient permissions). This means that the database’s structure could change while you are examining it in SQL Central.

When you first open a container, SQL Central reads its contents fresh from the database. If, however, someone then makes a change to the contents of that container, your view of it may be out of date. For example, if you are working in the **Tables** folder, and another user adds a table to the same database, your **Tables** folder will not immediately show the new table.

To ensure that you always have the latest changes to the database, SQL Central provides two **Refresh** commands in the **View** menu. These commands query the database to get up-to-date information:

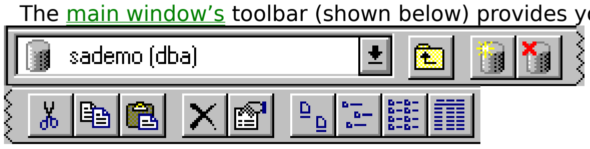
- **Refresh Folder** refreshes your view of the currently expanded container.
- **Refresh All** refreshes your view of all expanded containers.

---

Related Topics

## Using the main toolbar

The main window’s toolbar (shown below) provides you with graphic icons for common commands.



Click an icon above to see a description of what it does.

To show or hide the toolbar, click **Toolbar** in the **View** menu.

---

Related Topics

## Using the main status bar

In the main window, the status bar shows a brief summary of menu commands as you navigate through the menus.

To show or hide the status bar, click **Status Bar** in the **View** menu.

---

Related Topics

## Exiting from SQL Central

To quit SQL Central, click **Exit** in the **File** menu.

This disconnects all connected databases and exits the SQL Central application.

---

**Notes:**

- If you try to exit SQL Central while editing the code of a stored procedure, view, or trigger, SQL Central first prompts you to commit any unsaved changes.

---

Related Topics

## Showing all databases on a server

### To show all databases running on a given server that you are connected to:

- Open the desired server.
  {button ,AL(`server open',0,`',`')}   How?

This shows both connected databases and disconnected databases.

---

**Notes:**

- To connect to a disconnected database, see Connecting to a running database on a visible server.

---

Related Topics

## Starting a database without connecting

### To start a database on a visible server (without connecting to the database):

- Select the server, then do one of the following:
  - Click **Start Database** in the **File** menu.
  - Right-click one of the selected databases, then click **Start Database** in the pop-up menu.
- In the start dialog, do the following:
  - Enter a Database Name.
    {button ,PI(`',`IDH_DBX_database_name_details')} More details
  - Enter the path and name of a Database File.
    You can click **Browse** to locate the file using a standard dialog.
  - Click **OK** to start the database.

The database appears under the server as a disconnected database.

---

**Notes:**

- To connect to a database, see Connecting to a database.

---

Related Topics

## Stopping databases

### To stop one or more databases:

- Open the desired server.
- Select the databases, then do one of the following:
  - Click **Stop Database** in the **File** menu.
  - Right-click one of the selected databases, then click **Stop Database** in the pop-up menu.

This unconditionally stops the selected databases (even if there are connections to them).

---

### Notes:

- Stopping a connected database also disconnects you from it.
- This is functionally equivalent to the DBSTOP program.

---

Related Topics

## Showing performance statistics for a server

### To show performance statistics for a server:

- Open the desired server.
  {button ,AL(`server open',0,`',`')}   How?
- Open the **Statistics** folder.

The right panel shows the various statistics available for the server:

- Statistics that you've added to the performance monitor have this icon:
- Statistics not shown in the monitor have this icon:

---

**Notes:**

- You can also inspect statistics for connected users and for individual tables.

---

Related Topics

## Editing a statistic's properties

### To edit the properties of a server statistic:

- Open the **Statistics** folder.
- Open the statistic’s property sheet by doing one of the following:
  - Double-clicking the statistic
  - Selecting the statistic and clicking **Properties** in the **File** menu
  - Right-clicking the table, then clicking **Properties** in the pop-up menu
  - Selecting the statistic and clicking the **Properties** icon in the main toolbar

---

**Notes:**

- The only property you can change for a statistic is the graphing of the statistic in the SQL Central performance monitor.

---

Related Topics

## Overview of the SQL Central performance monitor

The SQL Central performance monitor displays a graph of statistics for a given SQL Anywhere server. The graph is shown in its own window, and is updated in real time (at adjustable intervals).

You can use the SQL Central monitor to graph statistics for any SQL Anywhere server that you can connect to in SQL Central.

Note, however, that the SQL Central monitor uses actual queries against the server to gather its statistics. This means that some statistics (such as Cache Reads/sec) are affected by the monitor itself.

As an alternative, you can graph server statistics using the Windows NT performance monitor.

---

Related Topics

## Using the SQL Central performance monitor

### To open the SQL Central performance monitor:

- Open the **Statistics** folder for the desired server.
- In the right panel, open the **Performance Monitor** template.
  (Note that you cannot open the monitor until you add statistics to it.)

To display particular statistics in the monitor, see Adding and removing statistics in the monitor.

If you close the monitor window, it will re-open automatically if you add more statistics to it.

### To display a continuous line graph that scrolls right to left as statistics are updated, do one of the following:

- In the monitor window, click **Line Graph** in the **View** menu.
- Click the **Line Graph** icon in the monitor toolbar.

### To display a stationary bar graph (where each bar shows that statistic's most recent value), do one of the following:

- In the monitor window, click **Bar Graph** in the **View** menu.
- Click the **Bar Graph** icon in the monitor toolbar.

To show or hide the legend (a detailed list of all graphed statistics), click **Legend** in the **View** menu of the monitor window.

To change the color or line style of a graphed statistic, edit the statistic's display properties.

To clear existing plotted values from the graph, click **Clear** in the **File** menu.

You can also set preferences for the monitor.

The monitor's status bar shows a brief summary of menu commands as you navigate through its menus.

---

Related Topics

## Adding and removing statistics in the monitor

### To add statistics to the SQL Central performance monitor, do one of the following:

- Open the **Statistics** folder.
- Do one of the following:
  - Select the statistics, then click **Add to Performance Monitor** in the **File** menu.
  - Right-click the statistics, then click **Add to Performance Monitor** in the pop-up menu.
  - Drag and drop the statistics onto the **Performance Monitor** template.

### To remove statistics from the monitor, do one of the following:

- In the **Statistics** folder, select the statistics, then click **Remove from Performance Monitor** in the **File** menu.
- In the **Statistics** folder, right-click the statistics, then click **Remove from Performance Monitor** in the pop-up menu.
- In the monitor window, select the statistics in the legend, then do one of the following:
  - Click **Remove Statistic** in the **Edit** menu.
  - Click the **Remove Statistic** icon in the monitor toolbar.

You can also add or remove a statistic in the monitor by opening the statistic's property sheet in the **Statistics** folder, then clicking the **Graph Statistic in SQL Central Performance Monitor** option.

---

Related Topics

## Editing a statistic's display properties in the monitor

**To edit the display properties of a server statistic in the SQL Central performance monitor:**

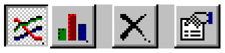
- In the monitor window, open the statistic's property sheet by doing one of the following:
  - Double-clicking the statistic in the legend
  - Selecting the statistic in the legend, then clicking **Properties** in the **Edit** menu
  - Selecting the statistic in the legend, then clicking the **Properties** icon in the monitor toolbar
- Select a **Color** and **Style** for the statistic.
- Click **OK** to save your changes and exit the dialog.

---

Related Topics

## Using the monitor toolbar

The SQL Central performance monitor toolbar (shown below) provides you with graphic icons for common commands.



Click an icon above to see a description of what it does.

To show or hide the toolbar, click **Toolbar** in the **View** menu.

---

Related Topics

## Setting preferences for the performance monitor

### To set your preferences for the SQL Central performance monitor:

- In the monitor window, click **Preferences** in the **File** menu to open the preferences dialog.
- In the **Interval** field, enter the number of seconds between updates.
  You can restore SQL Central's default interval by clicking **Default**.
- In the **Grid** section, select a spacing option for horizontal grid lines.
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- Setting very short intervals for updates can skew some statistics (like Cache Reads/sec) that are affected by the queries done by the performance monitor itself.

---

Related Topics

## Graphing with the Windows NT performance monitor

SQL Anywhere supports two ways to monitor your database servers:

- SQL Central's own integrated performance monitor
- integration with the Windows NT performance monitor (included with Windows NT)

If you can use the NT monitor (it can only be used in a NT-to-NT setup), you should use it instead of the SQL Central monitor because:

- The NT monitor offers more statistics (mainly those concerned with network communications).
- Unlike the SQL Central monitor, the NT monitor is non-intrusive. It uses a shared-memory scheme instead of performing queries against the server, so it does not affect the statistics themselves.

---

Related Topics

## Editing a database's properties

### To edit the properties of a database:

- Open the desired server.
- Open the database’s property sheet by doing one of the following:
  - Selecting the database and clicking **Properties** in the **File** menu
  - Right-clicking the database, then clicking **Properties** in the pop-up menu
  - Selecting the database and clicking the **Properties** icon in the main toolbar

---

**Notes:**

- The only properties you can change for a connected database are the SQL Remote publisher of the database and passthrough mode.
- Disconnected databases have an abbreviated property sheet. You cannot change the properties of a disconnected database.

---

Related Topics

## Showing system objects in a database

### To show or hide system objects in a database:

- Open the desired server.
- Select the connected database.
- Do one of the following:
  - Click **Show System Objects** in the **File** menu.
  - Right-click the database, then click **Show System Objects** in the pop-up menu.

The system tables, system views, system procedures, and system user-defined data types appear in their respective folders (for example, system tables appear alongside normal tables in the **Tables** folder).

---

Related Topics

## Connecting to a database in ISQL

### To open ISQL and automatically connect to a database:

- Open the desired server.
- Select the connected database.
- Do one of the following:
  - Click **Open ISQL** in the **File** menu.
  - Right-click the database, then click **Open ISQL** in the pop-up menu.

---

### Notes:

- If you already have an instance of ISQL running, SQL Central will use it instead of starting a new instance of ISQL.

---

Related Topics

## Validating a database

### To validate all indexes in a database:

- Open the desired server.
- Select the connected database, then do one of the following:
  - Click **Validate** in the **File** menu.
  - Right-click the database, then click **Validate** in the pop-up menu.

During the validation process, a status dialog shows the following:

- the names of the indexes as they are checked
- any validation errors encountered
- The total number of errors reported

---

**Notes:**

- You can also validate the indexes on a particular table instead of the entire database.
- This utility is functionally equivalent to the DBVALID program.

---

Related Topics

## Extracting from a consolidated database

### To extract a remote database from a consolidated database:

- Open the desired server.
- Select the consolidated database (it must be a connected database.
- Do one of the following:
  - Click **Extract Database** in the **File** menu.
  - Right-click the database, then click **Extract Database** in the pop-up menu.
  - Drag and drop the database onto the **Extract Database** wizard in the **Database Utilities** folder.
- Follow the instructions on each page of the wizard.

### To extract a remote database from any consolidated database (running or not):

- In the **Database Utilities** folder, open the **Extract Database** wizard.
- Follow the instructions on each page of the wizard.

For more information on SQL Remote replication, see Setting up a SQL Remote replication using SQL Central.

---

**Notes:**

- You can also invoke the **Extract Remote Database** wizard directly for a particular remote user.

---

Related Topics

## Examining or stopping passthrough mode for SQL Remote

### To examine the status of (or to stop) a passthrough from a consolidated database to remote databases:

- Open the desired server.
- Select the consolidated database. (It must be a connected database.
- Do one of the following:
  - Click **Passthrough** in the **File** menu.
  - Right-click the database, then click **Passthrough** in the pop-up menu.
  - Open the database's property sheet, then click **Passthrough** on the **SQL Remote** page.

This opens the passthrough dialog described in Using passthrough mode with remote databases.

---

Related Topics

## Disconnecting yourself from databases

**To disconnect from one or more connected databases, do one of the following:**

- Open the desired server.
- Select the databases, then do one of the following:
  - Click **Disconnect** in the **File** menu.
  - Right-click one of the selected databases, then click **Disconnect** in the pop-up menu.
- Click **Disconnect** in the **Tools** menu, or click the **Disconnect** icon in the main toolbar.
  This opens the disconnection dialog:
  - In the **Connections** list, select the databases.
  - Click **Disconnect**.

---

**Notes:**

- You can also disconnect other users from a given database.

---

Related Topics

## Creating a new database

### To create a new SQL Anywhere database:

- In the **Database Utilities** folder, open the **Create Database** wizard.
- Follow the instructions on each page of the wizard.

Once you've created a database and connected to it, you can start creating objects in it (tables, users, and so on).

---

**Notes:**

- This utility is functionally equivalent to the DBINIT program.

---

Related Topics

## Upgrading a database

### To upgrade an existing database (Watcom SQL 3.2 or 4.0, or SQL Anywhere 5) to the latest SQL Anywhere 5 database format:

- In the **Database Utilities** folder, open the **Upgrade Database** wizard.
- Follow the instructions on each page of the wizard.

---

### Notes:

- You cannot upgrade a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- The upgrade process works on the database that you specify, upgrading it **in place**.
  Because of this, we recommend that you back up a database before upgrading it.
- This utility is functionally equivalent to the DBUPGRAD program.

---

Related Topics

## Backing up a database

### To back up a connected database:

- Open the desired server.
- Select the database, then do one of the following:
  - Click **Backup** in the **File** menu.
  - Right-click the database, then click **Backup** in the pop-up menu.
  - Drag and drop the database onto the **Backup Database** wizard in the **Database Utilities** folder.
- Follow the instructions on each page of the wizard.

### To back up any database (running or not) or a write file:

- In the **Database Utilities** folder, open the **Backup Database** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- This utility is functionally equivalent to the DBBACKUP program.

---

Related Topics

## Compressing a database

### To compress a database into a compressed database file:

- In the **Database Utilities** folder, open the **Compress Database** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot compress a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBSHRINK program.

---

Related Topics

## Uncompressing a database

### To uncompress a compressed database file into a normal database file:

- In the **Database Utilities** folder, open the **Uncompress Database** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot uncompress a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBEXPAND program.

---

Related Topics

## Creating a write file

### To create a write file for an existing database:

- In the **Database Utilities** folder, open the **Create Write File** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot create a write file for a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBWRITE program.

---

Related Topics

## Translating a log file

**To translate an existing transaction log into a SQL command file:**

- In the **Database Utilities** folder, open the **Translate Log** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot translate a log for a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBTRAN program.

---

Related Topics

## Changing a log file

### To change the name of the transaction log file for a database or write file:

- In the **Database Utilities** folder, open the **Change Log File** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot change the name of a log for a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBLOG program.

---

Related Topics

## Unloading a database

### To unload a connected database:

- Open the desired server.
- Select the database, then do one of the following:
  - Click **Unload** in the **File** menu.
  - Right-click the database, then click **Unload** in the pop-up menu.
  - Drag and drop the database onto the **Unload Database** wizard in the **Database Utilities** folder.
- Follow the instructions on each page of the wizard.

### To unload any database or write file to a SQL command file:

- In the **Database Utilities** folder, open the **Unload Database** wizard.
- Follow the instructions on each page of the wizard.

---

### Notes:

- This utility is functionally equivalent to the DBUNLOAD program.

---

Related Topics

## Erasing a database

### To erase all or part of an existing database, write file, or compressed database file:

- In the **Database Utilities** folder, open the **Erase Database** wizard.
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot erase a database if you are already connected to it (in SQL Central or any other application). You must disconnect from the database first.
- This utility is functionally equivalent to the DBERASE program.

---

Related Topics

## Showing tables in a database

**To show the base tables and global temporary tables in a database:**

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **Tables** folder.

---

**Notes:**

- For a given database, you also can choose to show or hide system tables.

---

Related Topics

## Creating a new table

### To create a new base table or global temporary table in a database:

- Open the **Tables** folder.
- Open the table creation wizard by doing one of the following:
  - Opening the **Add Table** template in the right panel
  - Clicking **New > Table** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Table** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

**Notes:**

- Once you have created a new table, you can populate it with columns, a primary key, foreign keys, indexes, and triggers.

---

Related Topics

## Opening the table's property sheet

### To open the property sheet of a given table:

- Open the **Tables** folder.
- Open the table’s property sheet by doing one of the following:
  - Selecting the table and clicking **Properties** in the **File** menu
  - Right-clicking the table, then clicking **Properties** in the pop-up menu
  - Selecting the table and clicking the **Properties** icon in the main toolbar
- Make your changes to the following properties of the table:
  - general properties
  - primary key
  - unique constraints and the check constraint
  - permissions
  - table statistics
- Click **OK** to save your changes and exit the dialog.

---

**Notes:**

- You can also inspect statistics for the server and for connected users.

---

Related Topics

## Showing the primary key

The primary key of a table is shown in two places:

- in the **Columns** page of the table’s property sheet.
- in the **Columns** folder of the table.

In both places, the columns of the primary key are represented by primary-key icons to distinguish them from non-key columns.

The order of the columns in the primary key is shown in the **Primary Key Columns** field in the table’s property sheet.

### Notes:

- The lists in both the table property sheet and the **Columns** folder show the primary key columns (along with the non-key columns) in the order that they were created in the database. This may differ from the actual ordering of columns in the primary key.

Related Topics

## Editing the primary key

### To edit the primary key of a given table:

- Open the table’s property sheet.
- Click the **Columns** tab.
- To append a column to the primary key, select the column and click **Add To Key**.
- To remove a column from the primary key, select the column and click **Remove From Key**.
  To remove all columns from the primary key, click **Remove All**.
- To inspect a column’s properties, double-click it, or select it and click **Details**.

The order of columns in the primary key is shown in the **Primary Key Columns** field. This order reflects the order in which you added the columns to the key. To change the order of the columns, you must remove some or all of them and re-add them to the key.

You can also edit the primary key from the table’s **Columns** folder.
{button ,PI(`',`IDH_DBX_primary_key_editing_details')}   How?

---

Related Topics

## Editing the primary key in the main window

You can edit the primary key directly in the table's **Columns** folder in the main window:

- To add a column to the primary key, do one of the following:
  - Select the column, then click **Add to Primary Key** in the **File** menu.
  - Right-click the column and click **Add to Primary Key** in the pop-up menu.
- To remove a column from the key, do one of the following:
  - Select the column, then click **Remove from Primary Key** in the **File** menu.
  - Right-click the column and click **Remove from Primary Key** in the pop-up menu.

## Managing unique constraints for a table

### To manage unique constraints on a given table:

- Open the table’s property sheet.
- Click the **Constraints** tab.

### To create a new unique constraint on the table:

- In the **Unique constraints** section, click **New** to open the constraint dialog.
- To append a column to the constraint, select the column and click **Add**.
  To remove a column from the constraint, select the column and click **Remove**.
  To remove all columns from the constraint, click **Remove All**.
  To inspect a column’s properties, double-click it, or select it and click **Details**.
- Click **OK** to exit the constraint dialog.

### To remove a unique constraint on the table:

- In the **Unique constraints** section, select the constraint, then click **Remove**.

---

### Notes:

- The **Constraints** list shows true table constraints (containing more than one column) **as well as** column constraints (containing a single column). Adding a single-column constraint here is equivalent to setting the column's **Unique** property in its property sheet.
- You cannot edit an existing constraint. You must remove it and create a new one.

---

Related Topics

## Editing the check constraint for a table

### To edit the check constraint for a given table:

- Open the table’s property sheet.
- Click the **Constraints** tab.
- In the **Check constraint** text box, make your changes to the check constraint.
  This text box offers the same syntax highlighting and drag-and-drop features used in the SQL Central code editor.

---

Related Topics

## Granting/revoking permissions on a table

### To grant or revoke permissions on a given table:

- Open the table’s property sheet.
- Click the **Permissions** tab.

### To grant table permissions to users or groups:

- Click **Grant** to open the selection dialog.
- Select the desired users and groups, then click **Grant Permission** to exit the dialog.
- In the **Permissions** section, set the various permissions and grant options by clicking the appropriate check boxes.
  You can also grant column permissions for the **Reference**, **Select**, and **Update** permissions.

### To revoke table permissions for users or groups:

- Select the users and groups, then click **Revoke**.

---

### Notes:

- You can also assign permissions from the **Permissions** page of the user/group property sheet.
  To assign permissions to many users and groups at once, use the table's property sheet.
  To assign permissions to many tables at once, use the user's property sheet.
- You can quickly assign full permissions to a table by dragging and dropping users or groups onto it (or by dragging the table itself onto a user or group).
- You can also assign full permissions to a table by copying and pasting.
  {button ,PI(`',`IDH_DBX_table_permissions_copy_paste_details')}   How?

---

Related Topics

## Copying and pasting for table permissions

You can assign full permissions on a table by copying and pasting:

- Select the users and groups (or tables), then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the destination table (or user/group), then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Granting/revoking permissions on table columns

### To grant or revoke permissions on columns of a table:

- Open the **Permissions** page of the table's property sheet.
- Click the ... button for **Reference**, **Select**, or **Update** to open the column permissions dialog.
- Click **Grant permission on only these columns**.
- Select columns in the list.
- Set the **Column has permission** check box and **with grant option** check box appropriately.
- Click **OK** to exit the dialog.

### To grant permissions on all columns (instead of specific columns):

- Open the **Permissions** page of the table's property sheet.
- Click the ... button for **Reference**, **Select**, or **Update** to open the column permissions dialog.
- Click **Grant permission on all columns**.
- Click **OK** to exit the dialog.

---

Related Topics

## Showing columns

### To show the columns of a given table:

- In the **Tables** folder, open the desired table.
- Open the **Columns** folder.

The columns of a table are also shown on the **Columns** page of its property sheet.

---

Related Topics

## Creating a new column

### To create a new column in a given table:

- For the desired table, open the **Columns** folder.
- Open a new column property sheet by doing one of the following:
  - Opening the **Add Column** template in the right panel
  - Clicking **New > Column** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Column** in the pop-up menu
- On the **General** page, enter a name and optional comment for the new column.
- On the **Data type** page, set the column’s data type and advanced properties.
- Click **OK** to create the new column in the table.

---

**Notes:**

- You can also create columns in a table by copying them from other tables.

---

Related Topics

## Editing a column's properties

### To edit the properties of a given table column:

- For the desired table, open the **Columns** folder.
- Open the column's property sheet by doing one of the following:
  - Double-clicking the column
  - Selecting the column and clicking **Properties** in the **File** menu
  - Right-clicking the column, then clicking **Properties** in the pop-up menu
  - Selecting the column and clicking the **Properties** icon in the main toolbar
- On the **General** page, make your changes to the column's name or comment.
- On the **Data Type** page, make your changes to the column's data type and advanced properties.
- Click **OK** to save your changes and exit the dialog.

---

**Notes:**

- If you set a column's **Unique** property, the column will also appear as a single-column unique table constraint in the **Constraints** page of the table's property sheet.

---

Related Topics

## Setting the column's data type

The column's data type is shown on the **Data type** page of the column's property sheet.

### To set a column's data type:

- On the **Data Type** page, select a **Data type** from the pop-up list.
  The pop-up list includes both base data types and user-defined data types.
- Enter values for the **Size** or **Precision/Scale** of the data type (where applicable).

Selecting a user-defined data type **automatically** sets the column's default value, nullability, and check constraint if they are pre-defined by the user-defined data type.

On this page, you can also set the advanced properties of the column (default value, check constraint, unique constraint, and nullability).

---

Related Topics

## Setting advanced properties for a column

The advanced properties of a column (default value, check constraint, unique constraint, and nullability) are shown on the **Data type** page of the column's property sheet.

Selecting a user-defined data type sets the column’s data type, and also **automatically** sets the column's default value, nullability, and check constraint if they are pre-defined by the user-defined data type.

### To edit advanced properties of the column:

- Open the advanced properties dialog.
  {button ,PI(`',`IDH_DBX_column_properties_advanced_open')}   How?
- Enter an optional **User-defined** default value, or choose a **Pre-defined** default value from the pop-up list.
  If you've based this column on a user-defined data type, you can retain the type's default value (if any) or override it for this column.
- Set the column's unique constraint using the **Values are unique** option.
- Set the column's nullability using the **Type allows NULL** option.
  If you've based this column on a user-defined data type, you can retain the type's nullability (if any) or override it for this column.
- Enter an optional check constraint for the column.
  If you've based this column on a user-defined data type, you can retain the type's check constraint (if any) or override it for this column.
  This text box offers the same syntax highlighting and drag-and-drop features used in the SQL Central code editor.

---

Related Topics

## Opening the advanced properties dialog

To open the advanced properties dialog for a column, do one of the following:

- On the **Data type** page of the column’s property sheet, click **Edit**.
- In the main window, select the column and do one of the following:
    - Click **Advanced Properties** in the **File** menu.
    - Right-click the column, then click **Advanced Properties** in the pop-up menu.

## Copying columns within/between tables

To avoid having to re-create similar columns from scratch, you can copy columns within a table or between tables.

### To copy columns by dragging and dropping:

- Open the source table's **Columns** folder.
- In the right panel, select the columns to copy.
- If you're copying within the same table, hold down the **Ctrl** key.
- Drag and drop the columns onto the destination table or on its **Columns** folder.
- In the copy dialog that appears for each column, you can enter a name for the new column (you must do this if the column name conflicts with an existing column name in the destination table).
  Click **OK** to create the new column.

---

### Notes:

- The column's comment is **not** copied to the new column.
- You cannot copy a not-NULL column to a table that already contains rows of data.
  If you try to do this, you will get a "table must be empty" error.

---

Related Topics

## Deleting columns

### To delete one or more columns from a table:

- For the desired table, open the **Columns** folder.
- Select the columns to delete.
- Delete the columns by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected columns, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

Related Topics

## Showing foreign keys

**To show the foreign keys of a given table:**

- Open the desired table.
- Open the **Foreign Keys** folder.

---

Related Topics

## Creating a new foreign key

### To create a new foreign key in a given table:

- For the desired table, open the **Foreign Keys** folder.
- Open the foreign-key creation wizard by doing one of the following:
  - Opening the **Add Foreign Key** template in the right panel
  - Clicking **New > Foreign Key** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Foreign Key** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

### Notes:

- You can quickly create a foreign key in a given table by dragging and dropping the primary table onto it (both tables must be in the same database). This automatically opens the foreign-key creation wizard.
- You can also create a foreign key in a given table by copying and pasting.
  {button ,PI(`',`IDH_DBX_foreign_key_copy_paste_details')}   How?
- You cannot create a foreign key in a table if the table contains values for the foreign columns that can't be matched to values in the primary table's primary key.
  If you try to do this, you will get a "no primary key value for foreign key <name> in table <name>" error.

---

Related Topics

### Copying and pasting for foreign keys

You can create a foreign key in a given table by copying and pasting the primary table onto it:

- Select the primary table, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the foreign table, then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

This automatically opens the foreign-key creation wizard.

## Editing a foreign key's properties

### To edit the properties of a given foreign key:

- For the desired table, open the **Foreign Keys** folder.
- Open the foreign key’s property sheet by doing one of the following:
  - Double-clicking the foreign key
  - Selecting the foreign key and clicking **Properties** in the **File** menu
  - Right-clicking the foreign key, then clicking **Properties** in the pop-up menu
  - Selecting the foreign key and clicking the **Properties** icon in the main toolbar
- On the **General** page, make your changes to the foreign key’s name or comment.
- The **Columns** page shows the columns used and referenced in the foreign key.
  To inspect a column’s properties, double-click it, or select it and click **Details**.
- On the **Integrity** page, choose the appropriate **Update action** and **Delete action**, and set the appropriate **Advanced foreign key properties**.
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- You cannot change the columns in the foreign key except by deleting and recreating the key.
- You cannot set the **Set Null** action or **Allows Null** property for a foreign key if none of its columns have **Allow Nulls** set.
  Similarly, you cannot set the **Set default** action for a foreign key if its columns do not have default values defined.

---

Related Topics

## Deleting foreign keys

### To delete one or more foreign keys from a table:

- For the desired table, open the **Foreign Keys** folder.
- Select the foreign keys to delete.
- Delete the foreign keys by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected foreign keys, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar.

---

Related Topics

## Showing references from other tables

### To show, for a given table, which tables have foreign keys that reference it:

- Open the desired table.
- Open the **Referenced By** folder.

This shows any referencing (foreign) tables. Double-clicking a referencing table opens that table's property sheet.

---

Related Topics

## Showing indexes

### To show the indexes of a given table:

- Open the desired table.
- Open the **Indexes** folder.

---

Related Topics

## Creating a new index

### To create a new index for a given table:

- For the desired table, open the **Indexes** folder.
- Open the index creation wizard by doing one of the following:
  - Opening the **Add Index** template in the right panel
  - Clicking **New > Index** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Index** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

Related Topics

## Editing an index's properties

### To edit the properties of a given table index:

- For the desired table, open the **Indexes** folder.
- Open the index’s property sheet by doing one of the following:
  - Double-clicking the index
  - Selecting the index and clicking **Properties** in the **File** menu
  - Right-clicking the index, then clicking **Properties** in the pop-up menu
  - Selecting the index and clicking the **Properties** icon in the main toolbar
- On the **General** page, make your changes to the index’s name, comment, or **Unique** property.
- The **Columns** page shows the columns used in the index.
  To inspect a column’s properties, double-click it, or select it and click **Details**.
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- You cannot change the database space or columns used in the index except by deleting and recreating the index.

---

Related Topics

## Deleting indexes

### To delete one or more index from a table:

- For the desired table, open the **Indexes** folder.
- Select the indexes to delete.
- Delete the indexes by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected indexes, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar.

---

Related Topics

## Showing triggers

### To show the triggers of a given table:

- Open the desired table.
- Open the **Triggers** folder.

---

Related Topics

## Creating a new trigger

### To create a new trigger for a given table:

- For the desired table, open the **Triggers** folder.
- Open the trigger creation wizard by doing one of the following:
  - Opening the **Add Trigger** template in the right panel
  - Clicking **New > Trigger** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Trigger** in the pop-up menu
- Follow the instructions on each page of the wizard.
- When the wizard finishes and opens the code editor for you, complete the code of the trigger.
  For help, see Using the code editor.
- Execute the creation code by doing one of the following:
  - Clicking **Execute Script** in the **File** menu
  - Clicking the **Execute Script** icon in the editor toolbar

---

Related Topics

## Inspecting a trigger's code

### To inspect the code of a trigger:

- For the desired table, open the **Triggers** folder.
- Open the trigger in the code editor by doing one of the following:
  - Double-clicking the trigger
  - Selecting the trigger and clicking an **Open** command in the **File** menu
  - Right-clicking the trigger, then clicking an **Open** command in the pop-up menu

The **Open** commands work as follows:

- **Open** shows the trigger code in the SQL dialect in which it was last saved.
- **Open as Watcom-SQL** shows the trigger code in Watcom-SQL dialect.
- **Open as Transact-SQL** shows the trigger code in Transact-SQL dialect.

---

### Notes:

- To change or rename a trigger, you must create a new trigger with the new name, copy the old trigger code to the new trigger, then delete the old trigger.

---

Related Topics

## Editing a trigger's properties

### To edit the properties of a given trigger:

- For the desired table, open the **Triggers** folder.
- Open the trigger's property sheet by doing one of the following:
  - Selecting the trigger and clicking **Properties** in the **File** menu
  - Right-clicking the trigger, then clicking **Properties** in the pop-up menu
  - Selecting the trigger and clicking the **Properties** icon in the main toolbar

---

**Notes:**

- The only property you can change for the trigger is the comment.
- To change or rename a trigger, you must create a new trigger with the new name, copy the old trigger code to the new trigger, then delete the old trigger.

---

Related Topics

## Deleting triggers

### To delete one or more triggers from a table:

- For the desired table, open the **Triggers** folder.
- Select the triggers to delete.
- Delete the triggers by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected triggers, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar.

---

Related Topics

## Browsing a table's data in ISQL

### To open ISQL and browse a base table's data:

- In the **Tables** folder, select the base table to browse.
- Do one of the following:
  - Click **View Data** in the **File** menu.
  - Right-click the table, then click **View Data** in the pop-up menu.

---

**Notes:**

- If you already have an instance of ISQL running, SQL Central will use it instead of starting a new instance of ISQL.
- To browse the table, ISQL executes a **select * from <owner>.<table>** statement.
- You cannot use the **View Data** command on a global temporary table, since a given data set for the table is only available in the same connection that it was created in. (SQL Central uses its own connection, and you cannot create data directly in SQL Central).

---

Related Topics

## Copying tables within/between databases

**To avoid having to re-create similar tables from scratch, you can copy table definitions within a connected database or between databases:**

- To copy tables by dragging and dropping:
  - Hold down the **Ctrl** key.
  - Drag and drop the tables onto the **Tables** folder of the desired database.
- To copy tables by copying and pasting:
  - Select the first table, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
  - Select the **Tables** folder, then click an empty spot in the right panel.
  - Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

In the copy dialog that appears for each of these tables:

- Enter a name for the new table.
  The default name is the source table name with **_copy** appended.
- Enter the creator of the new table.
- Click **OK** to create the new table.

This creates the new table and copies the original table's columns to it.

### Notes:

- Only the table itself and its columns are copied. The other table properties (check constraints, etc.), child objects (indexes, etc.), and data are **not** copied.

Related Topics

## Validating tables

### To validate all indexes in a given set of tables:

- In the **Tables** folder, select the tables to validate.
- Validate the tables by doing one of the following:
  - Clicking **Validate** in the **File** menu
  - Right-clicking one of the selected tables, then clicking **Validate** in the pop-up menu

During the validation process, a status dialog shows the following:

- the names of the indexes as they are checked
- any validation errors encountered
- The total number of errors reported

---

**Notes:**

- You can also validate all indexes in a database at once.

---

Related Topics

## Deleting tables

### To delete one or more tables from a database:

- In the **Tables** folder, select the tables to delete.
- Delete the tables by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected tables, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar.

---

### Notes:

- You cannot delete a table that is being used as an article in a SQL Remote publication.
  If you try to do this, you will get a "table <name> has publications" error.

---

Related Topics

## Showing views in a database

### To show the views in a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}  How?
- Open the **Views** folder.

---

### Notes:

- For a given database, you can choose to show or hide system views.

---

Related Topics

## Creating a new view

### To create a new view in a database:

- Open the **Views** folder.
- Open the code editor by doing one of the following:
  - Opening the **Add View** template in the right panel
  - Clicking **New > View** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > View** in the pop-up menu
- Enter the code for the view.
  For help, see Using the code editor.
- Execute the creation code by doing one of the following:
  - Clicking **Execute Script** in the **File** menu
  - Clicking the **Execute Script** icon in the editor toolbar

---

Related Topics

## Inspecting a view's code

### To inspect the code of a view:

- Open the **Views** folder.
- Open the view in the code editor by doing one of the following:
  - Double-clicking the view
  - Selecting the view and clicking **Open** in the **File** menu
  - Right-clicking the view, then clicking **Open** in the pop-up menu

---

**Notes:**

- To change or rename a view, you must create a new view with the new name, copy the old view code to the new view, then delete the old view.

---

Related Topics

## Opening a view's property sheet

### To open the property sheet of a given view:

- Open the **Views** folder.
- Open the view’s property sheet by doing one of the following:
  - Selecting the view and clicking **Properties** in the **File** menu
  - Right-clicking the view, then clicking **Properties** in the pop-up menu
  - Selecting the view and clicking the **Properties** icon in the main toolbar
- Make your changes to the following properties of the view:
  - general properties
  - permissions
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- To rename a view, you must create a new view with the new name, copy the old view code to the new view, then delete the old view.

---

Related Topics

## Granting/revoking permissions on a view

### To grant or revoke permissions on a given view:

- Open the view’s property sheet.
- Click the **Permissions** tab.

### To grant view permissions to users or groups:

- Click **Grant** to open the selection dialog.
- Select the desired users and groups, then click **Grant Permission** to exit the dialog.
- In the **Permissions** section, set the various permissions and grant options by clicking the appropriate check boxes.

### To revoke view permissions for users or groups:

- Select the users and groups, then click **Revoke**.

---

### Notes:

- You can also assign permissions from the **Permissions** page of the user/group property sheet.
  To assign permissions to many users and groups at once, use the view's property sheet.
  To assign permissions to many views at once, use the user's property sheet.
- You can quickly assign full permissions to a view by dragging and dropping users or groups onto it (or by dragging the view itself onto a user or group).
- You can also assign full permissions to a view by copying and pasting.
  {button ,PI(`',`IDH_DBX_view_permissions_copy_paste_details')}   How?

---

Related Topics

## Copying and pasting for view permissions

You can assign full permissions on a view by copying and pasting:

- Select the users and groups (or views), then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the destination view (or user/group), then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Browsing a view's data in ISQL

### To open ISQL and browse a view's data:

- In the **Views** folder, select the view to browse.
- Do one of the following:
  - Click **View Data** in the **File** menu.
  - Right-click the view, then click **View Data** in the pop-up menu.

---

**Notes:**

- If you already have an instance of ISQL running, SQL Central will use it instead of starting a new instance of ISQL.
- To browse the view, ISQL executes a **select * from <owner>.<view>** statement

---

Related Topics

## Copying views within/between databases

**To avoid having to re-create similar views from scratch, you can copy view definitions within a connected database or between databases:**

- To copy views by dragging and dropping:
  - Hold down the **Ctrl** key.
  - Drag and drop the views onto the **Views** folder of the desired database.
- To copy views by copying and pasting:
  - Select the first view, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
  - Select the **Views** folder, then click an empty spot in the right panel.
  - Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

In the copy dialog that appears for each of these views:

- Enter a name for the new view.
  The default name is the source view name with **_copy** appended.
- Enter the creator of the new view.
- Click **OK** to create the new view.

This creates the new view and copies the original view's code to it.

---

### Notes:

- Only the view code is copied to the new view. The other properties of the view (permissions, etc.) are **not** copied.

---

Related Topics

## Deleting views

### To delete one or more views from a database:

- In the **Views** folder, select the views to delete.
- Delete the views by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected views, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

Related Topics

## Showing stored procedures in a database

### To show the stored procedures in a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')} How?
- Open the **Stored Procedures** folder.

---

### Notes:

- The **Stored Procedures** folder shows both procedures and functions (a special kind of procedure). For brevity, we use the term **procedure** to include both procedures and functions.
- For a given database, you can choose to show or hide system procedures.

---

Related Topics

## Creating a new stored procedure

### To create a new stored procedure in a database:

- Open the **Stored Procedures** folder.
- Open the code editor by doing one of the following:
  - Opening the **Add Procedure** template in the right panel
  - Clicking **New > Procedure** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Procedure** in the pop-up menu
- Enter the code of the stored procedure.
  For help, see Using the code editor.
- Execute the creation code by doing one of the following:
  - Clicking **Execute Script** in the **File** menu
  - Clicking the **Execute Script** icon in the editor toolbar

---

Related Topics

## Editing a stored procedure's code

### To edit the code of a stored procedure:

- Open the **Stored Procedures** folder.
- Open the procedure in the code editor by doing one of the following:
  - Double-clicking the procedure
  - Selecting the procedure and clicking an **Open** command in the **File** menu
  - Right-clicking the procedure, then clicking an **Open** command in the pop-up menu
- Edit the code of the stored procedure.
  For help, see Using the code editor.
- Save the updated procedure by doing one of the following:
  - Clicking **Execute Script** in the **File** menu
  - Clicking the **Execute Script** icon in the editor toolbar

The **Open** commands work as follows:

- **Open** shows the procedure code in the SQL dialect in which it was last saved.
- **Open as Watcom-SQL** shows the procedure code in Watcom-SQL dialect.
- **Open as Transact-SQL** shows the procedure code in Transact-SQL dialect.

---

### Notes:

- To rename a procedure, you must create a new procedure with the new name, copy the old procedure code to the new procedure, then delete the old procedure.

---

Related Topics

## Opening a stored procedure's property sheet

### To open the property sheet of a given stored procedure:

- Open the **Stored Procedures** folder.
- Open the procedure's property sheet by doing one of the following:
  - Selecting the procedure and clicking **Properties** in the **File** menu
  - Right-clicking the procedure, then clicking **Properties** in the pop-up menu
  - Selecting the procedure and clicking the **Properties** icon in the main toolbar
- Make your changes to the following properties of the procedure:
  - general properties
  - permissions
- Click **OK** to save your changes and exit the dialog.

---

**Notes:**

- To rename a procedure, you must create a new procedure with the new name, copy the old procedure code to the new procedure, then delete the old procedure.

---

Related Topics

## Granting/revoking permissions on a stored procedure

### To grant or revoke permissions on a given procedure:

- In the **Stored Procedures** folder, open the procedure’s property sheet.
- Click the **Permissions** tab.

### To grant permission to execute the procedure to users or groups:

- Click **Grant Execute** to open the selection dialog.
- Select the desired users and groups, then click **Grant Permission** to exit the dialog.

### To revoke permission to execute the procedure from users or groups:

- Select the users and groups, then click **Revoke Execute**.

---

**Notes:**

- You can also assign permissions from the **Permissions** page of the user/group property sheet.
  To assign permissions to many users and groups at once, use the procedure's property sheet.
  To assign permissions to many procedures at once, use the user's property sheet.
- You can quickly assign full permissions on a procedure by dragging and dropping users or groups onto it (or by dragging the procedure itself onto a user or group).
- You can also assign full permissions to a procedure by copying and pasting.
  {button ,PI(`',`IDH_DBX_procedure_permissions_copy_paste_details')}  How?

---

Related Topics

## Copying and pasting for procedure permissions

You can assign full permissions on a procedure by copying and pasting:

- Select the users and groups (or procedures), then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the destination procedure (or user/group), then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Testing a stored procedure in ISQL

### To test a stored procedure in ISQL:

- In the **Stored Procedures** folder, select the procedure to test.
- Do one of the following:
  - Click **Test Procedure** in the **File** menu.
  - Right-click the procedure, then click **Test Procedure** in the pop-up menu.

This opens ISQL and does the following:

- connects to the database
- in the command window, inserts a script containing commands to:
  - drop, declare, and set variables for the procedure's parameters
  - call the procedure

You can then set the variables to your desired values, execute the script, and view the results as you normally would in ISQL.

### Notes:

- If you already have an instance of ISQL running, SQL Central will use it instead of starting a new instance of ISQL.

Related Topics

## Copying stored procedures within/between databases

### To avoid having to re-create similar procedures from scratch, you can copy procedures within a connected database or between databases:

- To copy procedures by dragging and dropping:
  - Hold down the **Ctrl** key.
  - Drag and drop the procedures onto the **Stored Procedures** folder of the desired database.
- To copy procedures by copying and pasting:
  - Select the first procedure, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
  - Select the **Stored Procedures** folder, then click an empty spot in the right panel.
  - Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

In the copy dialog that appears for each of these procedures:

- Enter a name for the new procedure.
  The default name is the source procedure name with **_copy** appended.
- Enter the creator of the new procedure.
- Click **OK** to create the new procedure.

This creates the new procedure and copies the original procedure's code to it.

---

### Notes:

- Only the procedure code is copied to the new procedure. The other properties of the procedure (permissions, etc.) are **not** copied.

---

Related Topics

## Deleting stored procedures

### To delete one or more stored procedures from a database:

- In the **Stored Procedures** folder, select the procedures to delete.
- Delete the procedures by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected procedures, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

Related Topics

## Showing users and groups in a database

### To show the users and groups of a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **Users & Groups** folder.

The left panel shows all groups in the database (since it only shows containers). The right panel shows both users **and** groups.

You can open groups to see the users and groups that they contain.

---

### Notes:

- You can also show all users and groups currently connected to a database.

---

Related Topics

## Creating a new user or group

### To create a new user or group in a database:

- Open the **Users & Groups** folder.
- Open the **Add User** wizard or **Add Group** wizard by doing one of the following.
  - Opening the **Add User** or **Add Group** template in the right panel
  - Clicking **New > User** or **New > Group** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > User** or **New > Group** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

**Notes:**

- New users and groups are automatically added to the PUBLIC group.
- Once you have created a new user or group, you can:
  - add it to other groups
  - set its permissions on tables, views, and stored procedures
  - set it as the publisher or as a remote user of the database

---

Related Topics

## Opening a user/group's property sheet

### To open the property sheet of a user or group:

- Open the **Users & Groups** folder.
  For remote users, you can also open the **Remote Users** folder in the **SQL Remote** folder.
- Open the property sheet of the user or group by doing one of the following:
  - Double-clicking the user (note that double-clicking a group expands it instead)
  - Selecting the user or group and clicking **Properties** in the **File** menu
  - Right-clicking the user or group, then clicking **Properties** in the pop-up menu
  - Selecting the user or group and clicking the **Properties** icon in the main toolbar
- Make your changes to the following properties of the user or group:
  - general properties
  - authorities
  - membership in groups
  - permissions
  - SQL Remote properties (for remote users only)
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- You cannot change the name of a existing user or group. Instead, you can delete it and create a new user or group with the new name and same properties.

---

Related Topics

## Joining groups

**In the user/group property sheet, you can add the user or group to another group:**

- In the user/group property sheet, click the **Membership** tab.
- Click **Join Group** to open the selection dialog.
- Select the desired groups, then click **Join Group** to exit the dialog.

---

**Notes:**

- You can quickly add users or groups to other groups by dragging and dropping:
  - In the right panel, select the users and groups to add.
  - Drag the users and groups to the left panel and drop them onto the destination group.
- You can also add users or groups to other groups by copying and pasting them.
  {button ,PI(`',`IDH_DBX_user_group_join_copy_paste_details')} How?
- You cannot join a group twice. If you try to do this using drag-and-drop or copy/paste, you will get a "primary key for table 'SYSGROUP' is not unique" error.

---

Related Topics

## Joining groups by copying and pasting

You can add users or groups to other groups by copying and pasting them:

- Select the users and groups, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the destination group, then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Leaving groups

### In the user/group property sheet, you can remove the user or group from another group:

- In the user/group property sheet, click the **Membership** tab.
- Select the group to leave, then click **Leave Group**.

### You can also remove users/groups from another group directly from the group's container:

- In the **Users & Groups** folder, open the group that you want to remove your users or groups from.
- In the opened group, select the users and groups.
- Remove the selected users and groups by doing one of the following:
  - Clicking **Leave Group** in the **File** menu
  - Right-clicking one of the selected users or groups, then clicking **Leave Group** in the pop-up menu

---

**Notes:**

- Removing a user or group from a group does **not** delete them from the database (or from other groups). To do this, you must delete the user/group from the **Users & Groups** folder itself.

---

Related Topics

## Granting/revoking object permissions for a user/group

### To edit the permissions of a user or group on tables, views, or stored procedures:

- In the **Users & Groups** folder, open the property sheet of the user or group.
- Click the **Permissions** tab.
- Click the type of object to **View permissions on** (tables, views, or stored procedures).
  The list shows the permissions of the current user or group on the objects of that type.
- Select the objects to set permissions on.
- Set the various permissions and grant options by clicking the appropriate check boxes.

### To open the Permissions page for a given table, view, or stored procedure:

- In the list, select the table, view, or procedure.
- Click **Properties**.
  This opens the object’s property sheet and shows its **Permissions** page, ready for editing.

---

**Notes:**

- On the **Permissions** page for a user or group, you cannot set permissions on specific table columns. To do this, you must use the table's **Permissions** page (on its own property sheet).
- To assign permissions to many users and groups at once, use the object's property sheet.
  To assign permissions to many objects at once, use the user's property sheet.
- You can quickly assign full permissions on a given object by dragging and dropping users and groups onto it (or by dragging the object itself onto a user or group).
- You can also assign full permissions by copying and pasting.
  {button ,PI(`',`IDH_DBX_user_group_join_copy_paste_details')}   How?
- You should manage user permissions by assigning permissions to groups, then adding users to a group to give them the group’s permissions.
- If you're using SQL Remote replication, you can also grant and revoke publisher permissions and remote permissions for a user.

---

Related Topics

## Copying and pasting for user/group permissions

You can assign full permissions on a given table, view, or stored procedure by copying and pasting:

- Select the users and groups (or objects), then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the destination object (or user/group), then click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Deleting users and groups

### To delete users or groups from the database:

- In the **Users & Groups** folder, select the users and groups.
- Delete the selected users and groups by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected users or groups, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

**Notes:**

- Deleting a user or group also deletes all database objects (e.g. tables) that they own.
- You cannot delete users or groups when you select them in a group subfolder.
- Deleting users or groups from the database is different from removing them from other groups.
- Deleting a group from the database does **not** delete its members from the database, although they obviously lose membership in the deleted group.

---

Related Topics

## Showing user-defined data types in a database

### To show the user-defined data types in a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **User-defined Data Types** folder.

### Notes:

- For a given database, you can choose to show or hide system user-defined data types.

Related Topics

## Creating a new user-defined data type

### To create a new user-defined data type in a database:

- Open the **User-defined Data Types** folder.
- Open a new type property sheet by doing one of the following:
  - Opening the **Add User-defined Data Type** template in the right panel
  - Clicking **New > User-defined Data Type** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > User-defined Data Type** in the pop-up menu
- On the **General** page, enter a name for the new type.
- Click the **Advanced Options** tab.
- Choose a base data type, and enter any required parameters for it (size, precision, scale).
- Enter an optional **User-defined** default value, or choose a **Pre-defined** default value from the pop-up list.
- Set the user-defined data type's nullability using the **Type allows NULL** option.
- Enter an optional check constraint for the user-defined data type.
  This text box offers the same syntax highlighting and drag-and-drop features used in the SQL Central code editor.
- Click **OK** to create the new user-defined data type in the database.

Once you've create a user-defined data type, you can assign it to columns in the database:

---

Related Topics

## Editing properties of a user-defined data type

### To edit the properties of a user-defined data type in a database:

- Open the **User-defined Data Types** folder.
- Open the type's property sheet by doing one of the following:
  - Double-clicking the user-defined data type
  - Selecting the user-defined data type, then clicking **Properties** in the **File** menu
  - Right-clicking the user-defined data type, then clicking **Properties** in the pop-up menu
  - Selecting the user-defined data type, then clicking the **Properties** icon in the main toolbar

---

**Notes:**

- The only property you can change for the user-defined data type is the comment.
  To change other properties of the type, you must delete and recreate it.

---

Related Topics

## Deleting user-defined data types

### To delete one or more user-defined data types from a database:

- In the **User-defined Data Types** folder, select the types to delete.
- Delete the types by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected types, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

**Notes:**

- You cannot delete a user-defined data type that is being used by columns in the database.
  If you try to do this, you will get a "primary key for row in table <name> is referenced in another table" error.

---

Related Topics

## Setting up a SQL Remote replication using SQL Central

SQL Central allows you to set up a SQL Remote replication between a consolidated database and one or more remote databases. All setup is done at the consolidated database, usually before extracting the remote databases from it.

### To set up a SQL Remote replication using SQL Central:

- Connect to the consolidated database using a user account with DBA authority.
- Set the publisher of the consolidated database.
- For the message systems that you will be using (e.g. MAPI), edit the corresponding message types to include your publisher's address.
- Create new remote users, or grant remote permissions to existing users.
- Create publications based on the tables that you want to replicate.
- Subscribe the appropriate remote users to the appropriate publications.
- Extract the remote databases from the consolidated database using the extraction wizard.
- Deploy each remote database.
  {button ,PI(`',`IDH_DBX_SQL_remote_deploy_details')} How?

To help you maintain an existing SQL Remote replication setup, SQL Central supports SQL Remote's passthrough mode. This ensures that structural changes that you subsequently make to the consolidated database (such as changing a table's structure) are mirrored in its deployed remote databases.

For a full discussion of SQL Remote replication, see the SQL Anywhere User's Guide.

---

Related Topics

## Showing publications in a database

### To show the publications in a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **SQL Remote** folder.
- Open the **Publications** subfolder.

---

Related Topics

## Creating a new publication

### To create a new publication in a database:

- Open the **Publications** folder.
- Open the **Add Publication** wizard by doing one of the following.
  - Opening the **Add Publication** template in the right panel
  - Clicking **New > Publication** in the the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Publication** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

**Notes:**

- Once you have created a new publication, you can:
  - manage its articles
  - subscribe users to it

---

Related Topics

## Editing a publication's properties

### To edit the properties of a publication in a database:

- Open the **Publications** folder.
- Open the publication's property sheet by doing one of the following:
  - Selecting the publication, then clicking **Properties** in the **File** menu
  - Right-clicking the publication, then clicking **Properties** in the pop-up menu
  - Selecting the publication, then clicking the **Properties** icon in the main toolbar
- On the **General** page, you can change the publication's name and comment.
- On the **Articles** page, the list shows all articles in the publication, and any conflict triggers defined for the tables that those articles are based on.
  To create, edit, and delete articles, you must open the publication itself in the left panel of the main window.
- On the **Subscriptions** page, you can manage subscriptions for the publication.
- Click **OK** to save your changes and exit the dialog.

---

Related Topics

## Showing articles in a publication

### To show the articles in a publication:

- Open the **Publications** folder.
- Open the desired publication.

---

**Notes:**

- The articles of a publication are also shown on the **Articles** page of its property sheet.

---

Related Topics

## Creating a new article

### To create a new article in a publication:

- In the **Publications** folder, open the desired publication.
- Open a new article property sheet by doing one of the following:
  - Opening the **Add Article** template in the right panel
  - Clicking **New > Article** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Article** in the pop-up menu
- On the **Table** page, choose a table, then choose all or some of the table's columns.
- On the **Where restriction** page, enter the WHERE clause (if any) for restricting the rows in the article.
  This text box offers the same syntax highlighting and drag-and-drop features used in the SQL Central code editor.
- On the **Subscribe restriction** page, enter the SUBSCRIBE BY column or clause (if any) for restricting the rows in the article.
  The **Subscribe by clause** text box offers the same syntax highlighting and drag-and-drop features used in the SQL Central code editor.
- Click **OK** to create the new article in the publication.

---

### Notes:

- You can quickly create new articles in a given publication by dragging and dropping tables (from the same database) onto the publication. For each table, this automatically opens a new article property sheet with the table pre-selected.
- You can also create new articles by copying and pasting tables.
  {button ,PI(`',`IDH_DBX_SQL_remote_article_create_copy_paste_details')} How?

---

Related Topics

## Creating new articles by copying and pasting

You can create new articles by copying and pasting tables:

- Select the tables, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Open the destination publication, then click an empty spot in the right panel.
- Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

For each table, this automatically opens a new article property sheet with the table pre-selected.

## Editing an article's properties

### To edit the properties of an article:

- In the **Publications** folder, open the desired publication.
- Open the article's property sheet by doing one of the following:
  - Double-clicking the article
  - Selecting the article and clicking **Properties** in the **File** menu
  - Right-clicking the article, then clicking **Properties** in the pop-up menu
  - Selecting the article and clicking the **Properties** icon in the main toolbar
- After making your changes to the article’s properties, click **OK** to save your changes and exit the dialog.

---

Related Topics

## Editing properties of an article's table

### To edit the properties of the table that an article is based on, do one of the following:

- On the **General** page of the article's property sheet, click **Table Properties**.
- In the right panel of the main window, select the article and do one of the following:
  - Click **Table Properties** in the **File** menu.
  - Right-click the article, then click **Table Properties** in the pop-up menu.

This opens the table's property sheet, ready for editing.

---

Related Topics

## Deleting articles

**To delete one or more articles from a publication:**

- In the **Publications** folder, open the desired publication.
- Select the articles to delete.
- Delete the articles by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected articles, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

**Notes:**

- Deleting an article does **not** delete the table on which the article was based.

Related Topics

## Managing subscriptions for a publication

### To manage subscriptions by remote users to a publication:

- Open the publication's property sheet.
- Click the **Subscriptions** tab.
  The list shows all subscriptions to this publication (listed by remote user name, SUBSCRIBE BY value (if any), and status).

You can **subscribe a remote user** to the publication.
{button ,PI(`',`IDH_DBX_SQL_remote_pub_subscribe_details')} How?

You can **unsubscribe remote users** from the publication.
{button ,PI(`',`IDH_DBX_SQL_remote_pub_unsubscribe_details')} How?

You can also manually **start, stop, or synchronize** subscriptions to the publication.
{button ,PI(`',`IDH_DBX_SQL_remote_pub_subscribe_advanced_details')} How?

---

### Notes:

- You can quickly subscribe remote users to a publication by dragging and dropping them onto the publication. In the subscription dialog that appears for each user, enter a **Value** for the **Subscribe By** parameter (if any).
- You can also subscribe remote users by copying and pasting.
  {button ,PI(`',`IDH_DBX_SQL_remote_pub_subscribe_copy_paste_details')} How?

---

Related Topics

## Subscribing users to the publication

### To subscribe a remote user to the publication:

- Open the subscription dialog by doing one of the following:
  - On the **Subscriptions** page of the property sheet, click **Subscribe For**.
  - In the main window, select the publication and click **Subscribe For** in the **File** menu, or right-click the publication and click **Subscribe For** in the pop-up menu.
- Select a user from the list of all remote users in the database.
  (You can open the remote user's property sheet by clicking **Properties**.)
- Enter a **Value** for the **Subscribe By** parameter (if any).
- Click **Subscribe** to exit the dialog.

## Unsubscribing users from the publication

### To unsubscribe remote users from the publication:

- On the **Subscriptions** page, select the users from the list.
- Click **Unsubscribe**.

## Subscribing users by copying and pasting

You can subscribe remote users to a publication by copying and pasting them onto the publication:

- Select the users and groups, then click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the publication, then click an empty spot in the right panel.
- Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

This opens a subscription dialog for each user.

### Starting/stopping/synchronizing for publications

#### **To** manually **start, stop, or synchronize subscriptions to the publication:**

- On the **Subscriptions** page of the property sheet, select the desired remote users from the list.
  You must select users whose subscriptions have already been stored in the database. (You can click **Apply** to make sure of this.)
- Click **Advanced** to open the advanced options dialog.
- Click **Start Now**, **Stop Now**, or **Synchronize Now**.
  The subscriptions are affected as soon as you click one of these buttons.
- Click **Close** to exit the dialog.

Note that subsequently clicking **Cancel** in the publication's property sheet will **not** cancel your advanced actions on the subscriptions.

## Deleting publications

### To delete one or more publications from a database:

- In the **Publications** folder, select the publications to delete.
- Delete the publications by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected publications, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

**Notes:**

- If you delete a publication, all subscriptions to that publication are automatically deleted as well.

---

Related Topics

## Showing users of SQL Remote in a database

### To show the users involved in SQL Remote replication in a database:

- Open the desired database.  
  {button ,AL(`database open',0,`',`')}   How?
- Open the **SQL Remote** folder.  
  This folder contains the publisher of the database.
- Open the **Remote Users** subfolder.  
  This folder contains all remote users of the database.

---

### Notes:

- Users of SQL Remote are shown in both the **Users & Groups** folder (along with normal users) and by themselves in the **SQL Remote** folder.

---

Related Topics

## Setting the publisher of a database

### To set an existing user or group as the publisher of a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **SQL Remote** folder, then open the **Set Publisher** template.
  **or,**
  on the **SQL Remote** page of the database's property sheet, click **Change**.
- In the user/group selection dialog, select a user or group and click **OK**.
  (Note that the selection dialog only shows users and groups who are not already remote users.)

You can also set the publisher directly from the **Users & Groups** folder.
{button ,PI(`',`IDH_DBX_SQL_remote_publisher_direct_details')}   How?

You can **revoke publisher permission** from a publisher.
{button ,PI(`',`IDH_DBX_SQL_remote_publisher_revoke_details')}   How?

---

Related Topics

**Setting the publisher directly**

**If there is no current publisher for the database, you can set the publisher directly from the** Users & Groups **folder by doing one of the following:**

- Select a non-remote user or group, then do one of the following:
  - Click **Set As Publisher** in the **File** menu.
  - Right-click the user or group and click **Set As Publisher** in the pop-up menu.
- When creating a new user or group, select the **Publisher** option in the creation wizard.

### Revoking publisher permission

### To revoke publisher permission from the current publisher:

- Select the publisher in the **Users & Groups** folder or **SQL Remote** folder.
- Do one of the following:
  - Click **Revoke Publisher** in the **File** menu.
  - Right-click the publisher and click **Revoke Publisher** in the pop-up menu.

## Creating a new remote user

### To create a new remote user in a database:

- Open the **Remote Users** folder.
- Open the **Add Remote User** wizard by doing one of the following.
  - Opening the **Add Remote User** template in the right panel
  - Clicking **New > Remote User** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Remote User** in the pop-up menu
- Follow the instructions on each page of the wizard.

---

**Notes:**

- You cannot create a new remote user until you define at least one message type in the database.
- By default, remote users are created with remote DBA authority. Since this authority is required by the message agent for access to the remote database, you shouldn't revoke it.
- Once you have created a new remote user, you can:
  - add it to other groups
  - set its object permissions (on tables, views, and stored procedures)
  - subscribe it to publications
  - revoke its remote permissions

---

Related Topics

## Editing properties of a remote user

Remote users have the same properties as normal SQL Anywhere users or groups, and they also have additional SQL Remote properties.

### To edit the SQL Remote properties of a remote user:

- Open the user's property sheet.
  On the **Authorities** page, the **Remote DBA authority** is checked. Since this authority is necessary for most replication functions, you should not revoke it.
- Click the **SQL Remote** tab.
- The subscription list lets you manage the user's subscriptions to publications.
- Choose a **Message type** for communicating with the publisher.
  The pop-up list shows the message types currently defined for the database.
- Enter a **Remote address** for the remote user.
- Choose a replication frequency for sending to this remote user.
  {button ,PI(`',`IDH_DBX_SQL_remote_user_frequency_details')}   How?

The **Statistics** page of the remote user's property sheet shows the replication statistics for that user (for example, the date and time of the last replication message sent by the user).

---

Related Topics

## Choosing a replication frequency

Choose a replication frequency for sending from the publisher to this remote user:

- **Send then close** tells the publisher's agent to run once, sending all pending updates, then shut down. This means that the agent must be restarted each time the publisher wants to send updates.
  In most replication setups, this option is not used for sending from the consolidated publisher to the remote user.
- **Send every (hh:mm)** tells the publisher's agent to run continuously, sending updates to the remote user periodically. Enter the interval in hours and minutes (e.g. 01:30 for every 90 minutes).
- **Send daily at (hh:mm)** tells the publisher's agent to run continuously, sending updates to the remote user each day at the given time. Enter the time in 24-hour format (e.g. 22:00 for 10 p.m.).

## Granting remote permissions to users

### To grant remote permissions to normal users and groups (making them remote users in a SQL Remote replication setup):

- In the **Users & Groups** folder, do one of the following:
  - Select the users and groups and click **Set Remote** in the **File** menu.
  - Right-click the users and groups and click **Set Remote** in the pop-up menu.
  - Drag and drop the users and groups onto the **Remote Users** folder in the **SQL Remote** folder.
  - **Copy** and **Paste** the users into the **Remote Users** folder.
    {button ,PI(`',`IDH_DBX_SQL_remote_user_permission_copy_paste_details')} How?
- A dialog prompts you to set a **Message type**, **Remote address**, and replication frequency.
  For help on these options, see Editing properties of a remote user.
- Click **OK** to exit the dialog.

Once you have granted remote permissions to a user or group, you can:

- subscribe it to publications
- revoke its remote permissions

---

**Notes:**

- You cannot grant remote permissions to a user or group until you define at least one message type in the database.
- While you can grant remote permissions to a group, those remote permissions do **not** automatically apply to users in the group (unlike table permissions, for example). To do this, you must explicitly grant remote permissions to each user in the group.
  Otherwise, remote groups behave exactly like remote users (and are categorized as remote users).

---

Related Topics

## Granting remote permissions by copying and pasting

### To grant remote permissions to users and groups by copying and pasting:

- Select the users and groups.
- Click **Copy** (using the **Edit** menu, pop-up menu, or toolbar icon).
- Select the **Remote Users** folder, then click an empty spot in the right panel.
- Click **Paste** (using the **Edit** menu, pop-up menu, or toolbar icon).

## Revoking remote permissions from users

**To revoke remote permissions from remote users (making them normal users or groups):**

- Open the **Users & Groups** folder or the **Remote Users** folder in the **SQL Remote** folder.
- Do one of the following:
  - Select the remote users and click **Revoke Remote** in the **File** menu.
  - Right-click the remote users and click **Revoke Remote** in the pop-up menu.

---

**Notes:**

- Revoking remote permissions for a remote user automatically unsubscribes that user from all publications.

---

Related Topics

## Managing subscriptions for a remote user

### To manage subscriptions to publications for a remote user:

- Open the **Users & Groups** folder or the **Remote Users** folder in the **SQL Remote** folder.
- Open the remote user's property sheet.
- Click the **SQL Remote** tab.
  The list shows all subscriptions by this user (listed by publication name, SUBSCRIBE BY value (if any), and status).

You can **subscribe the remote user** to a publication.
{button ,PI(`',`IDH_DBX_SQL_remote_user_subscribe_details')} How?

You can **unsubscribe the remote user** from publications.
{button ,PI(`',`IDH_DBX_SQL_remote_user_unsubscribe_details')} How?

You can also manually **start, stop, or synchronize** subscriptions for the remote user.
{button ,PI(`',`IDH_DBX_SQL_remote_user_subscribe_advanced_details')} How?

---

**Notes:**

- You can quickly subscribe remote users to a publication by dragging and dropping them onto the publication. In the subscription dialog that appears for each user, enter a **Value** for the **Subscribe By** parameter (if any).
- You can also subscribe remote users by copying and pasting.
  {button ,PI(`',`IDH_DBX_SQL_remote_pub_subscribe_copy_paste_details')} How?

---

Related Topics

## Subscribing the user to a publication

### To subscribe the remote user to an existing publication:

- Open the subscription dialog by doing one of the following:
  - On the **SQL Remote** page of the property sheet, click **Subscribe To**.
  - In the main window, select the remote user and click **Subscribe To** in the **File** menu, or right-click the remote user and click **Subscribe To** in the pop-up menu.
- Select a publication from the list of all publications in the database.
  (You can open the publication's property sheet by clicking **Properties**.)
- Enter a **Value** for the **Subscribe By** parameter (if any).
- Click **Subscribe** to exit the dialog.

### Unsubscribing the user from publications

#### To unsubscribe the remote user from existing publications:

- On the **SQL Remote** page of the property sheet, select the publications from the list.
- Click **Unsubscribe**.

### Starting/stopping/synchronizing for remote users

#### **To** manually **start, stop, or synchronize subscriptions for the remote user:**

- On the **SQL Remote** page, select the desired publications from the list.
  You must select publications whose subscriptions have already been stored in the database. (You can click **Apply** to make sure of this.)
- Click **Advanced** to open the advanced options dialog.
- Click **Start Now**, **Stop Now**, or **Synchronize Now**.
  The subscriptions are affected as soon as you click one of these buttons.
- Click **Close** to exit the dialog.

Note that subsequently clicking **Cancel** in the remote user's property sheet will **not** cancel your advanced actions on the subscriptions.

## Extracting a database for a remote user

### To extract a remote database for a particular remote user:

- Open the consolidated database.
  {button ,AL(`database open',0,`',`')}  How?
- Open the **Users & Groups** folder, or the **Remote Users** folder in the **SQL Remote** folder.
- Do one of the following:
  - Select the remote user and click **Extract Database** in the **File** menu.
  - Right-click the remote user and click **Extract Database** in the pop-up menu.
- Follow the instructions on each page of the wizard.

For more information on extracting databases, see Starting a SQL Remote replication.

---

**Notes:**

- You can also invoke the **Extract Remote Database** wizard directly from a consolidated database.

---

Related Topics

## Deleting users of SQL Remote

You cannot delete a publisher or remote user directly from the SQL Remote folder - you can only revoke their publisher permission or remote permission (respectively).

To delete a publisher or remote user from a database, you must select it in the main **Users & Groups** folder and delete it from there.

---

Related Topics

## Showing message types in a database

### To show the message types for SQL Remote replication in a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **SQL Remote** folder.
- Open the **Message Types** subfolder.

---

Related Topics

## Creating a new message type

### To create a new message type for SQL Remote replication in a database:

- Open the **Message Types** folder.
- Open a new message-type property sheet by doing one of the following:
  - Opening the **Add Message Type** template in the right panel
  - Clicking **New > Message Type** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > Message Type** in the pop-up menu
- Enter the name of the new message type. This name corresponds to the actual message system. The name that you enter should correspond to a message-type DLL already installed in your SQL Anywhere directory. (For example, you can enter **FOOBAR** if you have a **DBFOOBAR.DLL** available.)
- Enter a **Publisher address**. This is the address used by the remote database to send replication messages back to the publisher.
- Enter an optional **Comment**.
- Click **OK** to create the new message type in the database.

---

Related Topics

## Editing properties of a message type

### To edit the properties of a message type for SQL Remote replication:

- Open the **Message Types** folder.
- Open the message type's property sheet by doing one of the following:
  - Selecting the message type and clicking **Properties** in the **File** menu
  - Right-clicking the message type, then clicking **Properties** in the pop-up menu
  - Selecting the message type and clicking the **Properties** icon in the main toolbar
- Make your changes to the **Publisher address** or **Comment**.
- Click **OK** to save your changes and exit the dialog.

---

### Notes:

- You cannot change the name of a existing message type. Instead, you must delete it and create a new message type with the new name.

---

Related Topics

## Deleting message types

### To delete one or more message types from a database:

- Open the **Message Types** folder.
- Select the message types to delete.
- Delete the message types by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected message types, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

**Notes:**

- You cannot delete a message type if it is being used by a remote user in the database.
  If you try to do this, you will get a "message type <name> not found" error.

---

Related Topics

## Starting a SQL Remote replication (extracting)

Once you have set up your publications and users for SQL Remote, and subscribed them appropriately, you can extract the remote databases from the consolidated database and start the replication.

### To extract a remote database from the consolidated database:

- Open the consolidated database.
  {button ,AL(`database open',0,`',`')}   How?
- In the **Database Utilities** folder, open the **Extract Remote Database** wizard.
- Follow the instructions on each page of the wizard.

SQL Central then completes the extraction process.
{button ,PI(`',`IDH_DBX_SQL_remote_extract_details')}   How?

You can then deploy the remote database.
{button ,PI(`',`IDH_DBX_SQL_remote_deploy_details')}   How?

For more help, see the SQL Anywhere User's Guide.

---

### Notes:

- If you use the wizard to extract a non-running database, it will only be able to unload the structure and data for you. It will not be able to create the remote database and reload it.
  For this reason, we recommend that you always extract from a consolidated database that you are connected to in SQL Central.
- You can also invoke the extraction wizard for a particular database or for a particular remote user - SQL Central automatically fills in the appropriate database and/or user entries in the wizard.
- The extraction wizard always extracts (synchronizes) the remote database using the WITH SYNCHRONIZATION option. In those rare cases where you don't want to use this option, you must use the DBXTRACT command-line utility instead.

---

Related Topics

## SQL Central's extraction process

When you complete the extraction wizard, it does the following on your machine:

- creates the remote database
- extracts (unload) the relevant structures and/or data from the consolidated database to files
- loads those files into the newly created remote database

## Deploying a remote database

Once you’ve extracted a remote database from the consolidated database, you can then deploy the remote database by:

- moving it to a remote machine (via network or removable media)
- ensuring that the remote machine is able to communicate with the consolidated machine

## Using passthrough mode with remote databases

### To passthrough SQL statements from a consolidated database to remote databases in an existing SQL Remote replication setup:

- Open the consolidated database.
  {button ,AL(`database open',0,`',`')} How?
- Select the remote users for the passthrough
  **or,**
  select the publications whose subscribers should get the passthrough.
- Open the passthrough dialog by doing one of the following:
  - Selecting the users or publications and clicking **Passthrough** in the **File** menu
  - Right-clicking the users or publications, then clicking **Passthrough** in the pop-up menu
- In the dialog, the passthrough list shows all users and publications currently in passthrough mode (indicated by their **On** state), and also shows the users or publications that you want to add to the passthrough (indicated by their **Off** state).
  Make sure that these additional users or publications are selected.
- Click **Start**.
- If no existing users or publications were in passthrough mode when you opened the dialog, you can choose the **Only reflect changes in remote database** option (PASSTHROUGH ONLY mode).
- Click **OK** to add the selected objects to the passthrough.

When passthrough mode is active, SQL Central appends **Passthrough** to the consolidated database's name in the left and right panels of the main window.

### To stop passthrough mode for all remote users and publications of a consolidated database:

- Open the passthrough dialog as described above, or by opening it directly for a database.
- In the passthrough dialog, click **Stop All**.
- Click **OK** to stop passthrough for all remote users.

---

Related Topics

## Showing spaces in a database

### To show the database spaces for a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **DB Spaces** folder.

---

Related Topics

## Creating a new database space

### To create a new database space in a database:

- Open the **DB Spaces** folder.
- Open a new space property sheet by doing one of the following:
  - Opening the **Add DB Space** template in the right panel
  - Clicking **New > DB Space** in the **File** menu
  - Right-clicking an empty spot in the right panel, then clicking **New > DB Space** in the pop-up menu
- Enter the **Name** of the new space.
- Enter the **Filename** of the database file for the new space.
- Click **OK** to create the new space in the database.

---

Related Topics

## Editing a space's properties

### To edit the properties of a database space:

- Open the **DB Spaces** folder.
- Open the space’s property sheet by doing one of the following:
  - Selecting the space and clicking **Properties** in the **File** menu
  - Right-clicking the space, then clicking **Properties** in the pop-up menu
  - Selecting the space and clicking the **Properties** icon in the main toolbar
- If you've renamed or moved the database file, change the **Filename** of the space to point to it.
- To add pages beyond the current size, click **Add Pages**.
  In the pages dialog, enter the number of pages to add to the space.
- Click **OK** to save your changes and exit the dialog.

---

Related Topics

## Deleting database spaces

### To delete one or more database spaces from a database:

- Open the **DB Spaces** folder.
- Select the spaces to delete.
- Delete the spaces by doing one of the following:
  - Clicking **Delete** in the **File** menu
  - Right-clicking one of the selected spaces, then clicking **Delete** in the pop-up menu
  - Clicking the **Delete** icon in the main toolbar

---

**Notes:**

- Before you can delete a space, you must delete all tables that use that space.

---

Related Topics

## Showing users connected to a database

### To show all users connected to a database:

- Open the desired database.
  {button ,AL(`database open',0,`',`')}   How?
- Open the **Connected Users** folder.

This shows all other users currently connected to a given database (**not** including yourself), regardless of the client that they used to connect (SQL Central, ISQL, a custom client application, etc.).

---

### Notes:

- If more than one user is connected to the database under the same user name, you can distinguish them (to disconnect them, for example) by their connection ID.
  The connection ID is shown as a column in the **Details** view of the right panel, and is also shown in the property sheet of the connected user.
- You can also show all users and groups that have permission to connect to a database.

---

Related Topics

## Inspecting a connected user's properties

### To inspect the properties of a user's current connection to a database:

- Open the **Connected Users** folder.
- Open the connected user's property sheet by doing one of the following:
  - Double-clicking the connected user
  - Selecting the connected user and clicking **Properties** in the **File** menu
  - Right-clicking the connected user, then clicking **Properties** in the pop-up menu
  - Selecting the connected user and clicking the **Properties** icon in the main toolbar

The **General** page shows general information about the user's connection.

The **Statistics** page shows a list of statistics for the user's connection.

---

### Notes:

- You can also inspect statistics for the server and for individual tables.

---

Related Topics

## Disconnecting connected users

### To disconnect other users from a database:

- Open the **Connected Users** folder.
- Select the connected users to disconnect.
- Do one of the following:
  - Click **Disconnect** in the **File** menu.
  - Right-click one of the selected users, then click **Disconnect** in the pop-up menu.

---

**Notes:**

- You can also disconnect yourself (in SQL Central) from a given database.

---

Related Topics

## Using the code editor

SQL Central’s code editor is a separate window for displaying and editing the code of triggers, stored procedures, and views.

Beyond the standard text-editing functions, it provides:

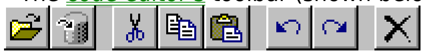
- syntax highlighting for both Watcom-SQL and Transact-SQL:
  - keywords and constants are shown in blue
  - quoted strings are shown in purple
  - partially quoted strings are shown in red
  - comments are shown in green
- unlimited **Undo** and **Redo**
- drag-and-drop editing (you can drag selected text to a new location in the code)
- a toolbar and status bar
- ability to open from and save to external files, and to execute the code against the database.

---

Related Topics

## Using the editor toolbar

The code editor's toolbar (shown below) provides you with graphic icons for common commands.



Click an icon above to see a description of what it does.

To show or hide the toolbar, click **Toolbar** in the **View** menu.

---

Related Topics

## Using the editor status bar

In the editor window, the status bar shows a brief summary of menu commands as you navigate through the menus.

To show or hide the status bar, click **Status Bar** in the **View** menu.

---

Related Topics

## Summary of drag-and-drop operations

In SQL Central, you can do the following tasks by dragging and dropping:

- adding statistics to the performance monitor
- extracting, backing up, and unloading a database
- copying columns within/between tables
- creating foreign keys between tables
- copying tables, views, and stored procedures within/between databases
- adding users and groups to other groups
- setting user permissions for tables, views, and stored procedures
- creating new articles in a publication
- subscribing remote users to a publication
- granting remote permissions to users
- moving text around in the code editor

---

Related Topics

## Specifying a database name

If you don’t enter a database name, the root of the database file is used.

For example, the database for the **SADEMO.DB** file would be named **SADEMO** by default.

## SQL Anywhere help is not available

The main help file for Sybase SQL Anywhere (**dbeng50w.hlp**) is not installed.

Some OEM versions of SQL Anywhere (including the version bundled with PowerBuilder) do not include this file.

Please refer to your printed SQL Anywhere manuals, or in the case of PowerBuilder, refer to the PowerBuilder on-line documentation.

**No related topics**

**Designed and written by Dave O'Brien**

Last revised: 03/04/96 4:20 PM

## Go to a different folder

Shows the current folder, and allows you to move up in the object tree by choosing a higher-level folder.

More Help

## Up One Level

Allows you to Moves up one level in the object tree (for example, from a given table to the **Tables** folder).

More Help

## Connect

Connects to a SQL Anywhere database.

More Help

## Disconnect

Disconnects from one or more connected databases.

More Help

## Cut

Places the selected objects in the clipboard. The selected objects are not removed from their current location until they are pasted elsewhere.

**Copy**

Places a copy of the selected objects in the clipboard.

## Paste

Pastes the objects in the clipboard into the current container. If the objects were **Cut**, they are then removed from their original location.

## Delete

Deletes the selected objects from the database.

## Properties

Opens the selected object’s property sheet.

## Large Icon

Shows objects in the right panel as large icons, arranged left to right, top to bottom.

More Help

## Small Icon

Shows objects in the right panel as small icons, arranged left to right, top to bottom.

More Help

## List

Shows objects in the right panel as small icons, arranged in as many columns as necessary.

More Help

## Details

Shows objects in the right panel as small icons, top to bottom, with columns for name, type, and other properties.

More Help

## Open From File

Loads text from an external file (for example, a text file containing SQL commands), replacing any existing text in the editor.

More Help

## Save To File

Saves the text in the editor to an external text file.

More Help

## Execute Script

Executes the code in the code editor as DDL against the database’s definition.

For example, when editing an existing procedure, clicking **Execute Script** would run the ALTER PROCEDURE statement against the database’s definition, thereby updating that procedure.

**Note: Execute Script** does not actually execute the procedure itself against data.

More Help

## Cut

Removes the selected text from the editor and places it in the clipboard.

## Copy

Places a copy of the selected text in the clipboard.

## Paste

Pastes the text in the clipboard into the editor at the cursor.

## Undo

Reverses the effect of your most recent change to the text. You can undo your actions in succession back to the original state of the text when you entered the editor.

### Redo

Used immediately after an **Undo** command, this reverses the effect of the **Undo**. If you have used **Undo** several times in succession, you can also **Redo** these changes in succession.

### Clear

Removes the selected text from the editor (without affecting the clipboard).

## Line Graph

Displays a continuous line graph that scrolls right to left as statistics are updated.

More Help

## Bar Graph

Displays a stationary bar graph where each bar shows that statistic’s most recent value.

More Help

## Remove Statistic

Removes the selected statistic from the performance monitor.

More Help

## Properties

Opens the selected statistic’s display property sheet.

More Help

A container that only contains other objects, and has no inherent properties of its own.

In SQL Central, a database object that holds other database objects.

Some containers (such as tables) are actual objects in themselves (as well as holding other objects). Other containers (such as the **Columns** folder in a table) are only folders - they don’t have any inherent properties of their own.

In SQL Central, a hierarchy of database objects, organized by server and database.

The object tree shows all servers that you have connected to (more precisely, the ones with databases that you have connected to in SQL Central).

For each server shown, you can see **all** databases running on it. You may be connected to some of these in SQL Central (called connected databases), and not connected to others (called disconnected databases).

Under each connected database, you can see many types of database objects (tables, views, procedures, and so on).

A database that you are connected to in SQL Central.

A connected database is represented by the icon, and shows its contents as an object tree under the database.

See also disconnected database.

A running database that is visible in SQL Central (that is, running on a visible server), but that you are not connected to in SQL Central.

A connected database is represented by the  icon.

Most database operations in SQL Central require you to be connected to the database. You cannot see the object tree under a database until you connect to it using SQL Central.

See also connected database.

A named set of connection parameters (user name, password, database name, and so on).

A special dialog, usually organized as a series of pages, that guides you through the steps of a complex task.

In the right panel, a special icon that performs a task.

Most templates automate the creation of new objects of a certain type. For example, the **Add Index** template opens a wizard that helps you create an index.

A dialog that presents the properties of a selected object. The properties are often arranged on several tabbed pages, and can usually be changed.

A real-time graph that shows performance statistics from any running SQL Anywhere server.

The color and line style of a server statistic as shown in the performance monitor.

Click an object using the right mouse button (instead of the left mouse button).

Closes this dialog without saving your changes.

Closes this dialog and saves your changes.

Closes this dialog.

Opens a standard file dialog to help you locate a file.

Provides a place for you to type the password of the SQL Anywhere user.
For security, the typed characters are shown as asterisks.

The full name of the current object.

The type of object.

Examples: table, primary key, user

The database user who created (and owns) this object.

Allows you to select a user to be the creator of the new object.

Shows all users defined in the database.

Exits the dialog and sets the selected user as the creator of the new object.

The table that this object belongs to.

Provides a place for you to type a comment (text description) of this object.

For example, you could use this area to describe the object’s purpose in the system.

Shows a summary of the properties of the selected objects.

Opens the property sheet of the selected object.

Allows you to grant permissions on this object to other users.

Exits this dialog and grants permissions to the selected users.

Revokes permissions on this object from the selected users.

The SUBSCRIBE BY column or clause.

If the articles in the publication use SUBSCRIBE BY columns that have different names, the column of the first article is displayed, followed by the other column names in brackets, separated by vertical bars.

Example: **province ( prov | territory )**

Provides a place for you to type the value of the SUBSCRIBE BY column or clause for the remote user.

The name of the subscription that you are starting, stopping, or synchronizing.

The current state of the subscription (**Started** or **Not started**).

Provides options to start subscriptions manually.

Starts the subscription manually.

Provides options to stop subscriptions.

Stops the subscription (if it has already been started).

Provides options to synchronize subscriptions manually.

Synchronize the subscription manually.

The name of the server to start the database on.

Starts the database on the server.

Brief explanation of the performance statistic.

Adds or removes the current statistic to/from the performance monitor window.

Provides options for setting the update interval of the monitor window.

Provides a place for you to type the number of seconds between updates in the monitor window.

Resets the update interval to the default of 1 second.

Provides options for setting the horizontal grid lines in the monitor window.

Hides the grid lines in the monitor window.

Draws a horizontal grid line at the 50% interval in the monitor window.

Draws horizontal grid lines at 25% intervals in the monitor window.

Draws horizontal grid lines at 10% intervals in the monitor window.

Allows you to select a color for plotting the current statistic.

Allows you to select a line style for plotting the current statistic.

Shows how the statistic will look using the selected color and style.

Shows all remote users and publications currently in passthrough mode (indicated by their **On** state). Also shows any remote users or publications that you selected to add to the passthrough (indicated by their **Off** state).

More Help

Sets the selected remote users and publications to start passthrough mode (when you click **OK** to exit this dialog).

More Help

Sets the database to stop passthrough mode for all of its objects (when you click **OK** to exit this dialog).

More Help

Sets a new passthrough mode to PASSTHROUGH ONLY (changes are only reflected in the remote databases, not in the consolidated database).

More Help

The version number of the SQL Anywhere database. (Versions before 5.0 are Watcom SQL databases.)

The root database file for the database (also represented by the SYSTEM database space).

The transaction log file for the database.

The transaction log mirror file for the database.

The page size of the database (in bytes).

Determines whether this database is encrypted.

Determines whether this database ignores trailing blanks in comparisons.

Determines whether this database is case-sensitive.

The default collation of the database.

A unique number that identifies the individual connection for the user (in this case, SQL Central’s connection to the database).

The total number of current connections to this database from all users (including SQL Central’s connection).

The publisher of the database.

Allows you to select a user or group to be the database’s publisher.

Shows the non-remote users of the current database.

Exits the dialog and sets the selected user or group as the publisher of the database.

The consolidated database of this database (if this database is acting as a remote database).

The number of remote users subscribing to publications in this database.

The number of subscriptions by remote users to publications in this database.

The number of subscriptions in this database that have been started.

Allows you to control passthrough mode for the database.

The database space used by the table.

Shows whether the rows of the global temporary table are deleted or preserved when a COMMIT is executed.

Shows the columns defined for the table (listed by column name and data type).
Primary-key columns are represented by different icons.

Adds the selected columns to the primary key of the table.

Removes the selected columns from the primary key of the table.

Removes all columns from the primary key of the table.

The data type of the column.

The default value of the column.

The comment for the column.

Shows whether the values of this column in the table’s rows must be unique.

Shows the nullability of the column.

The order of columns in the primary key.

Provides options for setting unique constraints for the table.

Shows the unique constraints defined for the table.

Allows you to create a new unique constraint for the table.

Shows all columns defined in the table (listed by column name and data type).

Adds the selected columns to the unique constraint (in list order).

Removes the selected columns from the unique constraint.

Removes all columns from the unique constraint.

The order of columns in the unique constraint.

Removes the selected unique constraint from the table.

Provides options for setting check constraints for the table.

Provides a place for you to type the check constraint on the table.
This text box offers the same syntax highlighting and drag-and-drop features used in the code editor.

Shows all users who have permissions on the table.
Permissions: **A**=Alter, **D**=Delete, **I**=Insert, **R**=Reference, **S**=Select, **U**=Update
Permissions that also have a grant option are marked with an asterisk (*).
More Help

For users in the list, those permissions that also have a grant option are marked with an asterisk (*).
More Help

Shows the types of permissions for the selected users.
More Help

Permits the selected users to alter the table’s structure or to create triggers for the table.
More Help

Permits the selected users to delete rows in the table or view.
More Help

Permits the selected users to insert rows in the table or view.
More Help

Permits the selected users to create indexes and foreign keys on the table.
More Help

Permits the selected users to read rows in the table or view.
More Help

Permits the selected users to update columns of the rows in the table or view.
More Help

For the selected users, shows whether the type of permission applies to all columns or a subset of columns.
For subsets of columns, those columns with grant option are marked with an asterisk (*).
More Help

Allows you to apply the permission to all columns or a subset of columns.
More Help

Grants this permission to all columns of the table (even if columns are subsequently added to the table).
More Help

Grants this permission to the selected columns of the table.
More Help

Shows the columns of the table (listed by name, permission state, and grant option state).
More Help

Grants this permission to the selected columns.
More Help

For this permission, sets the grant option for the selected columns.
More Help

Shows the grant option for each type of permission for the selected users .
More Help

For this permission, sets the grant option for the selected users .
More Help

The number of columns in the table.

The number of rows in the table at the time of the last checkpoint.

The number of bytes required for each row in this table.

This is calculated from the length of the string columns, the precision of **numeric** columns, and the number of bytes of storage for all other data types.

If the table includes **long binary** or **long varchar** columns, their arbitrary widths are **not** included, so the row width can only be approximated.

Provides options for setting the data type of the column.

Allows you to select the data type of the column.
The list includes base data types followed by user-defined data types.
New columns default to **integer**.

Provides a place for you to type the precision for the **numeric** data type.

Provides a place for you to type the scale for the **numeric** data type.

Provides a place for you to type the size for certain data types.

Describes the default value, check constraint, unique constraint, and nullability of the column.

The default value of the column.

If the column is based on a user-defined data type, it inherits the type’s default value (if any), but this can be overridden for the column.

Click **Edit** to change this for the column.

The check constraint of the column.

If the column is based on a user-defined data type, it inherits the type’s check constraint (if any), but this can be overridden for the column.

Click **Edit** to change this for the column.

Determines whether the values of this column in the table’s rows must be unique.
Click **Edit** to change this for the column.

The nullability of the column.

If the column is based on a user-defined data type, it inherits the type’s nullability (if any), but this can be overridden for the column.

Click **Edit** to change this for the column.

Allows you to change the advanced properties (default value, check constraint, unique constraint, and nullability) of the column.

Provides options for setting the default value of the column.

Allows you to type a custom value (string or number) for the default value.

Allows you to select a pre-defined value (for example, **current date**) for the default value.

Provides options for setting the column’s other advanced properties.

Determines whether this column’s value in each row must be unique.

Determines the nullability of this column.

Provides a place for you to type the check constraint on the column.
This text box offers the same syntax highlighting and drag-and-drop features used in the code editor.

The table containing the foreign key (this table).

The table containing the primary key that the foreign key is referencing.

Shows the columns in this table’s foreign key.

Shows the columns in the primary table’s primary key.

Provides options for setting the update action of the foreign key.

Prevents updates of the associated primary table’s primary key value if there are corresponding foreign keys in this table.

Updates the foreign key to match a new value for the associated primary key.

Sets to NULL all the foreign-key values in this table that correspond to the updated primary key of the associated primary table.

Sets to the column’s default value all the foreign-key values in this table that correspond to the updated primary key of the associated primary table.

Provides options for setting the delete action of the foreign key.

Prevents deletion of the associated primary table’s primary key value if there are corresponding foreign keys in this table.

Deletes the rows from this table that match the deleted primary key of the associated primary table.

Sets to NULL all the foreign-key values in this table that correspond to the deleted primary key of the associated primary table.

Sets to the column’s default value all the foreign-key values in this table that correspond to the deleted primary key of the associated primary table.

Provides options for setting the advanced properties of the foreign key.

Determines the nullability of the foreign-key columns.

Forces the database to wait for a COMMIT before checking the integrity of the foreign key, overriding the setting of the WAIT_FOR_COMMIT database option.

This option can only be used with the **Restrict** actions.

The database space used by the index.

Ensures that there will not be two rows in the table with identical values in all columns of the index.

Shows all columns in the table (listed by column name and data type).
Primary key columns are represented by a special icon.

The columns used in this index.
ASC=ascending column values, DESC=descending column values

Determines whether the trigger executes **Before** or **After** the event.

Row-level triggers can also have **SQL Remote conflict** timing, which executes before **UPDATE** or **UPDATE OF column-lists** events.

Determines which events cause the trigger to execute.
Events: **Insert**, **Delete**, **Update**, **Update Columns**.

Determines whether the trigger is row-level or statement-level.

For triggers in this table that execute for the same kind of event with the same timing, this number determines the order in which these triggers are fired.

Shows all users who have permissions on the view.
Permissions: **D**=Delete, **I**=Insert, **S**=Select, **U**=Update
Permissions that also have a grant option are marked with an asterisk (*).
More Help

Shows all users who have permission to execute the stored procedure.

For confirmation, this provides a place for you to re-type the new password that you entered in the **Password** field. The contents of the two fields must match exactly.

Determines whether the user or group is allowed to connect to the database.

If the user or group is not allowed to connect, the password (if any) is removed from the account. If you later change the user or group to allow them to connect, you must supply a new password.

Users are almost always allowed to connect. For a group, however, turning this option off prevents anyone from connecting to the database using the group account itself.

Grants DBA authority to the user or group.

Grants resource authority to the user or group.

For SQL Remote replication, grants remote DBA authority to the user or group.

Shows the groups to which the user or group belongs.

Allows you to add the user or group to other groups.

Shows all groups in the database.

Exits this dialog and adds the user or group to the selected groups.

Allows you to remove the user or group from the selected groups.

Allows you to select a type of object (table, view, or stored procedure) to show user permissions for.
More Help

Shows all objects of the given type in the database.
Permissions: **A**=Alter, **D**=Delete, **I**=Insert, **R**=Reference, **S**=Select, **U**=Update
Permissions that also have a grant option are marked with an asterisk (*).
More Help

Shows the types of permissions for the selected objects.
More Help

Permits the user or group to alter the structure of the selected objects, or to create triggers for the selected tables.

More Help

Permits the user or group to delete rows in the selected tables or views.
More Help

Permits the user or group to insert rows in the selected tables or views.
More Help

Permits the user or group to create indexes and foreign keys on the selected tables.
More Help

Permits the user or group to read rows in the selected tables or views.
More Help

Permits the user or group to update all columns of the rows in the selected tables or views.
More Help

Permits the user or group to execute the selected stored procedures.
More Help

Shows the grant option for each type of permission for the selected objects.
More Help

For this permission, sets the grant option for the selected objects.
More Help

Shows all subscriptions by this remote user (listed by publication name, SUBSCRIBE BY value (if any), and status).

More Help

Allows you to subscribe the remote user to an existing publication.
More Help

The name of the remote user who is subscribing.

Shows all publications in the database.

Exits this dialog and subscribes the remote user to the selected publication.

Removes the remote user’s subscriptions to the selected publications.
More Help

Allows you to manually start, stop, or synchronize a selected subscription by the remote user.

**Note:** You must select a subscription that has already been stored in the database. (You can click **Apply** to make sure of this.)

More Help

Allows you to select a message type defined in the database.
More Help

Provides a place for you to type the remote address of the remote user.
More Help

Sets the replication frequency so that the publisher’s agent will run once, sending all pending messages, then shut down.

This means that the agent must be restarted each time the publisher wants to send messages.

In most replication setups, this option is not used for sending from the consolidated publisher to the remote user.

More Help

Sets the replication frequency so that the publisher’s agent will run continuously, sending messages to the remote user at the given periodic interval.

More Help

Provides a place for you to type a periodic interval (in hours and minutes) as the replication frequency.
You can also use the spin buttons to change the values of each field.
Example: **01:30** (for every 90 minutes)
More Help

Sets the replication frequency so that the publisher’s agent will run continuously, sending messages to the remote user each day at the given time.

More Help

Provides a place for you to type a daily time (in 24-hour format) for the replication frequency.
You can also use the spin buttons to change the values of each field.
Example: **22:00** (for 10 p.m.)
More Help

Shows whether the remote user is also a consolidated database for other remote users (a multi-tier replication setup).

The date and time when this remote user will next send a

~

Allows you to select the base data type of the user-defined data type.

New user-defined data types default to **integer**.

Provides options for setting the default value of the user-defined data type.

Determines the nullability of columns based on this type.

Provides a place for you to type the check constraint for the user-defined data type.
This text box offers the same syntax highlighting and drag-and-drop features used in the code editor.

Shows all articles in the publication (listed by article name, article type, and conflict trigger (if any)).

Shows all subscriptions to this publication (listed by remote user name, SUBSCRIBE BY value (if any), and status).

Allows you to subscribe an existing remote user to the publication.

The name of the publication to subscribe to.

Shows all remote users in the database.

Exits this dialog and subscribes the selected remote user to the publication.

Removes the selected remote users’ subscriptions to the publication.

Allows you to manually start, stop, or synchronize a selected subscription to the publication.

**Note:** You must select a subscription that has already been stored in the database. (You can click **Apply** to make sure of this.)

The publication that this article belongs to.

The type of article (**Table** or **Table Subset**).

Opens the property sheet for the table that the article is based on.

The table that the article is based on.
For new articles, you can choose from a list of tables in the database.

Provides options for selecting the table columns that the article will use.

Sets the article to use all columns in the table.

Sets the article to use the table columns that you select in the list.

Shows all columns in the table that the article is based on.

Provides a place for you to type the WHERE clause to restrict the table rows that are included in the article. This text box offers the same syntax highlighting and drag-and-drop features used in the code editor.

Sets the article to **not** use SUBSCRIBE BY columns or clauses to partition rows.

Sets the article to partition rows from the table based on a column (SUBSCRIBE BY columns).

Allows you to select a table column for partitioning the rows from the table.

Sets the article to partition rows from the table based on an expression (SUBSCRIBE BY clause).

Provides a place for you to type the SUBSCRIBE BY expression for partitioning the rows from the table. This text box offers the same syntax highlighting and drag-and-drop features used in the code editor.

The name of the message type.

This name corresponds to the actual message system, and must be unique for all message types in the database.

The name should correspond to a message-type DLL already installed in your SQL Anywhere directory. For example, you could type **FOOBAR** if you have a **DBFOOBAR.DLL** available.

Provides a place for you to type the address used by the remote database to send replication messages back to the publisher.

Provides a place for you to type the name of the database file that the database space points to, in case you've renamed or moved the file.

If you don’t supply a path, the directory of the SYSTEM database space is assumed.

Example: **C:\TESTDB\ORDERS.DB**

Allows you to pre-allocate storage in the database space by adding pages to it.
This may improve performance for bulk-loading operations.

Provides a place for you to type the number of pages to add to the database space.
**Note:** Once you have added pages to the database space, you cannot remove them.

The optional name of the user’s connection.

Naming your connections allows multiple connections to the same database, or multiple connections to the same or different database servers, all simultaneously.

The connection ID of the user’s connection.
This number can be useful for distinguishing between users connected under the same user name.
More Help

A unique number identifying the database to which the user is connected.

The full path and filename (that is, the SYSTEM database space) of the database to which the user is connected.

The full path and filename of the transaction log file of the database to which the user is connected.

The type of communications link used by the user’s connection. If the connection is between a SQL Anywhere client and network server, the link type represents the network protocol being used.

Examples: **LOCAL**, **IPX**

The communications port ID used by the user’s connection.

The number of users currently connected to this SQL Anywhere server (including your current SQL Central connection).

Shows a list of statistics for the user’s connection (listed by statistic name, value, and description).

Provides a place for you to type the name of the SQL Anywhere user .

More Help

Provides a place for you to type the name of the SQL Anywhere stand-alone engine or network server.

Example: **myserver**
More Help

Provides a place for you to type the database name.

If you are starting a database, you can specify a new name for the database to run as.
If you don’t enter a name, it defaults to the database file name without the ".DB" extension.
Example: **WSAMPLE** (for the **WSAMPLE.DB** sample database)
More Help

Provides options for starting a stand-alone engine or network server.

More Help

Provides a place for you to type the full path and name of the SQL Anywhere database file or write file.
Example: **C:\SQLANY50\WSAMPLE.DB**
More Help

Starts the database file using a SQL Anywhere stand-alone engine.

More Help

Starts a SQL Anywhere client that connects to the database specified by **Database Name** on the network server specified by **Server Name**.

More Help

Starts the database with command-line parameters that you supply.
To specify these parameters, click the **Custom** command button.
More Help

Opens a dialog to specify custom startup parameters for the database.

More Help

Expands this dialog to reveal more options for connecting to a database.

More Help

Connects to the specified database using the options chosen in this dialog.

More Help

Closes this dialog without connecting to a database.

More Help

Provides a place for you to type the command and switches for starting the database.

- stand-alone engine:

  **dbeng50** <stand-alone engine switches>

  For example, to open the sample database using an engine with a cache size to 5 megabytes, you would type: **dbeng50 -c 5M sademo.db**

- SQL Anywhere client to network server:

  **dbclient** <client switches>
  <network server switches>

  For example, to connect a client to a server using TCP/IP as the network protocol, you would type: **dbclient -x tcpip my_server**

Shuts down the database when the last connection to it is terminated. This is different from the **-ga** engine switch (which auto-stops the engine itself).

Shows all current database connections for your SQL Central session.

Disconnects the selected connection (confirmation required), then exits this dialog.

Shows all currently defined connection profiles, and whether they are automatically connected each time that SQL Central is started.

Shortcut: Double-click a profile to **Connect**.

More Help

Tries to connect using the parameters of the selected profile.

More Help

Closes this dialog.

More Help

Allows you to create a new connection profile.

More Help

Provides a place for you to type the name of the new profile.

Example: **sademo as dba**

Allows you to select the type of database.

Types: **Sybase SQL Anywhere**

Accepts the new profile name and opens the connection parameter dialog.

Edits the parameters of the selected profile.

More Help

Removes (deletes) the selected profile (confirmation required).

More Help

Toggles the automatic-startup option for the selected profile.

When the automatic-startup option is turned on, the profile is automatically connected each time that SQL Central is started.

More Help

Closes this dialog.

Can also shift this program’s control into an alternate state…

## About the glossary

This glossary contains words used in the SQL Anywhere User's Guide and industry-wide terms concerning SQL databases and related technologies.

address
In SQL Remote replication, the destination of replication messages sent by a given message system.   Publishers and remote users each have their own address.

The format of the address depends on the message type.

article
In SQL Remote replication, a database object that represents a whole table, or a subset of the rows and columns in a table.   Articles are grouped together in publications.

authority
Determines what structural actions a user can perform in a database.   While most users will have no special authorities, a user with DBA authority can grant other users resource authority, DBA authority, or remote DBA authority.

autocommit
An ISQL option. If autocommit is set to TRUE, then a database COMMIT is performed after each successful command and a ROLLBACK after each failed command. The ODBC interface has a setting similar to autocommit. Users of applications communicating with SQL Anywhere through ODBC should check their application documentation for how to set this option. Developers programming directly to the ODBC interface should consult the ODBC documentation for information about implementing autocommit.

backup
It is important to make regular backups of your database files in case of media failure.   A backup is a copy of the database file.   You can make backups using the SQL Anywhere backup utility or using other archiving software of your choice.

base data type
One of the intrinsic (simple) data types included in SQL Anywhere (such as INTEGER).   User-defined data types (including those supplied with SQL Anywhere) are built on base data types.

base table
The tables that permanently hold the data in the database are sometimes called **base tables** to distinguish them from temporary tables and from views.

batch
A batch is a set of SQL statements sent together to the database engine by an application.

buffer
An area of memory reserved for some dedicated use, such as storing incoming messages for processing.

business rules
The rules to which the data in a database must conform are often called business rules, as they reflect organizations' operating practices.

For example, a business rule limiting customers' credit can be enforced by a trigger in a sales order table.   A rule requiring library books to be returned within two weeks can be checked by constraints, so that reminders can be issued or fines calculated.

cache

To avoid having to access a hard disk every time it needs to retrieve or write information to the database, SQL Anywhere keeps data it may need to access again in the computer's memory, where access is much quicker.   The area of memory set aside for this information is called a cache.   See also checkpoint log.

case sensitivity
When creating a database, you can choose whether identifiers in this database (table names, column names, and so on) and values are considered to be case-sensitive in comparisons and string operations.   If, for example, a database is case-sensitive, emp_id would not be the same as emp_ID.

checkpoint
A time at which all dirty pages held in cache are written to disk is called a checkpoint.   Once all the dirty pages are written to disk, the checkpoint log is deleted.   A checkpoint may occur for one of several reasons, at times specified by the user or decided internally by the database engine.

checkpoint log
A SQL Anywhere database file is composed of pages.   Before a page is updated (made dirty), a copy of the original is always made.   The copied pages are the checkpoint log.   The checkpoint log is deleted when a checkpoint occurs.

client
Client is a widely-used term with several meanings.   It refers to the user's side of a client/server arrangement:   for example, an application that addresses a database, typically held elsewhere on a network, is called a client application.

code page
A code page is a character set. SQL Anywhere supports many different national languages and character sets, as long as they are available under the user's operating system. See also: collation.

collation
SQL Anywhere supports many different national languages by providing a choice of collations. Each collation specifies a code page (character set) and a sorting and comparison order for that code page. SQL Anywhere also supports custom collations. See also: code page.

collation sequence
A collation sequence is an ordering of characters used for sorting and comparing strings.   Each national language employs a character set or code page, and a collation sequence for that set.

column
All data in relational databases such as SQL Anywhere is held in tables, composed of rows and columns. Each column holds a particular type of information. See also: table, row.

command file
A text file containing SQL statements.  Command files can be built by yourself (manually) or by database utilities (automatically).  The DBUNLOAD utility, for example, creates a command file consisting of the SQL statements necessary to recreate a given database.

command window
The ISQL command window is an edit control for entering SQL statements for execution.

comment
A text description of an object in the database.   These remarks are not parsed or executed by the database engine.

commit
When a user sends the SQL command COMMIT, all the work done to that point is applied to the database itself, and can no longer be undone.   A commit marks the end of a transaction.   See also:   transaction, rollback.

compound statement
A compound statement is a set of SQL statements treated as a unit. The body of a procedure or trigger consists of a compound statement. A compound statement starts with BEGIN and finishes with END. See also: stored procedure, trigger.

compressed database file
A database file that has been compressed to a smaller physical size using SQL Anywhere's database compression utility (DBSHRINK).   To make changes to a compressed database file, you must use an associated write file. Compressed database files can be re-expanded into normal database files using SQL Anywhere's database uncompression utility (DBEXPAND).

concurrency
Multi-user versions of SQL Anywhere support concurrent applications:   separate connections which may address the same data in the database, running at the same time.   SQL Anywhere provides transaction processing and automatic row-level locking to ensure that information remains consistent and that each concurrent application sees a consistent set of data.   See also:   transaction, locking.

conflict trigger
In SQL Remote replication, a trigger that is fired when an update conflict is detected, before the update is applied. Specifically, conflict triggers are fired by the failure of values in the VERIFY clause of an UPDATE statement to match the values in the database before the update.   They are fired before each row is updated.

connection
When a client application connects to a database, it specifies several parameters that govern all aspects of the connection once it is established.   A user ID, a password, the name of the database to attach to, are all parameters that specify the connection.   All exchange of information between the client application and the database to which it is connected is governed by the connection.   See also:   user ID, password, named connection.

connection ID
A unique number that identifies a given connection between the user and the database.   You can determine your own connection ID using the following SQL statement inside that connection:

```
        select connection_property( 'Number' )
```

connection parameters
When a client application connects to a database, the connection parameters specify the characteristics of the connection.   See connection.

connection string
When a client application connects to a database, it uses connection parameters to specify the characteristics of the connection.  These connection parameters are collected together as a connection string.

See also:  connection parameters.

consolidated database
In SQL Remote replication, a database that serves as the "master" database in the replication setup.   The consolidated database contains all of the data to be replicated, while its remote databases may only contain their own subsets of the data.   In case of conflict or discrepancy, the consolidated database is considered to have the primary copy of all data.

constraint
When tables and columns are created they may have constraints assigned to them. A constraint ensures that all entries in the database object to which it applies satisfy a particular condition. For example, a column may have a UNIQUE constraint, which requires that all values in the column be different. A table may have a foreign key constraint, which specifies how the information in the table relates to that in some other table.

See also: integrity, foreign key constraint, primary key constraint, column, table. unique constraint, check constraint.

container
In a graphical user interface, a container is an object that contains other objects.   Containers can be expanded by double-clicking them.

cursor
A cursor is a handle, or identifier, for a particular SQL query and for the position within the result set that is being accessed.   Cursors allow each row of a query that returns more than one row to be processed by a client application individually.

cursors
A cursor is a handle, or identifier, for a particular SQL query and for the position within the result set that is being accessed.   Cursors allow each row of a query that returns more than one row to be processed by a client application individually.

cursor stability
Cursor stability is a concurrency condition, guaranteed by choosing an isolation level of 1, 2, or 3.   Cursor stability guarantees that no row fetched through a cursor yields uncommitted data.

daemon
A background process on a computer, running periodically or all the time, which manages a particular function such as printing services or network communication services.

data dictionary
See system tables.

data source
Databases available to ODBC applications are defined through data sources.

data type
Each column in a table is associated with a particular data type. Integers, character strings, and dates are examples of data types.

database
A relational database is a collection of tables, related by primary and foreign keys.   The tables hold the information in the database, and the tables and keys together define the structure of the database.   A database may be stored in one or more database files, on one or more devices.

database administrator
The database administrator (DBA) is a person responsible for maintaining the database. The DBA is generally responsible for all changes to a database schema, and for managing users and user groups.

The role of database administrator is built in to SQL Anywhere databases as a user ID. When a database is initialized, a DBA user ID is created. The DBA user ID has authority to carry out any activity within the database.

database connection
All exchange of information between client applications and the database takes place in a particular connection. A valid user ID and password are required to establish a connection, and the actions that can be carried out during the connection are defined by the privileges granted to the user ID.

database engine
All access to information in a SQL Anywhere database goes through a SQL Anywhere engine.   The specific SQL Anywhere engine you are using will depend on your operating system.   Requests for information from a database are sent to the database engine, which carries out the instructions.

database file
A database is held in one or more distinct database files.   The user does not have to be concerned with the organization of a database into files:   requests are issued to the database engine about a database, and the engine knows in which file to look for each piece of required information.

Each table, together with its associated indexes, must be contained in a single database file.

database name
When a database is loaded by an engine, it is assigned a database name.   Client applications can connect to a database by specifying its database name.

The default database name is the root of the database file.

database object
A database is made up of tables, indexes, views, procedures, and triggers.   Each of these is a database object.

database owner
The user ID that creates a database is the owner of that database, and has the authority to carry out any changes to that database.   The database owner is also referred to as the database administrator, or DBA.   A database owner can grant permission to other users to have access to the database and to carry out different operations on the database, such as creating tables or stored procedures.

datagram
Communications across a network may take place in a session, (also called a connection, or virtual circuit) or in a connectionless manner.   In the connectionless case, the independent packets of information are called datagrams, in analogy with telegrams.

Connectionless communications require routing decisions for all packets, and are not guaranteed.   Multi-user editions of SQL Anywhere server use datagrams for their TCP/IP, IPX, and NetDG communication links.

DBA
An abbreviation for database administrator, also called the database owner.   When a database is first created, using the DBINIT utility, it is created with the single user ID **DBA**, with password **SQL**.

DBA authority
DBA (DataBase Administrator) authority enables a user to carry out any activity in the database (create tables, change table structures, assign ownership of new objects, create new users, revoke permissions from users, and so on).   The DBA user has DBA authority by default.

dbspace
A SQL Anywhere database can be held in multiple files, called dbspaces.   The SQL command CREATE DBSPACE adds a new file to the database.

Each table, together with its associated indexes, must be contained in a single database file.

DDE
Dynamic data exchange (DDE) is a method for Windows applications to communicate with each other.   In any DDE conversation, one application is the client, or destination, while the other application is the server, or source.

default value
Also known as a "column default" or just "default", this is a value that is automatically assigned to particular columns when a new row is entered into a database table, without any action on the part of the client application, as long as no value is specified by the client application.   If the client application does specify a value for the column, it overrides the column's default value.   Default values can be user-defined (a string or number) or pre-defined (e.g.   a timestamp supplied by the system).

device
A device is a disk drive, a tape drive, or other information storage medium.

DLL

A dynamic link library, or DLL, is a collection of compiled functions that can be addressed by a running application at run time. DLL's allow a single set of functions to be shared by many applications, and can be updated without updating every application that depends on them. The Windows, Windows NT, and OS/2 operating systems support DLLs.

driver
A piece of software that manages low-level functions on a computer, such as a communication with a network card or a printer.

Embedded SQL
The native programming interface to SQL Anywhere from C programs. SQL Anywhere embedded SQL is an implementation of the ANSI and IBM standard.

encryption
Encryption makes it more difficult for someone to decipher the data in your database by using a disk utility to look at the file.   File-compression utilities are not able to compress an encrypted database file as much as an unencrypted one.

entity integrity
Each row in every table in a database must be uniquely identifiable in order to be accessed by the database engine. Each row is identified by its primary key. This requirement of identifiability is called entity integrity, and is automatically ensured by SQL Anywhere through a checking of the primary key. See also: primary key, referential integrity.

engine name
When a database engine is started it is assigned an engine name.   Client applications specify the engine to which they want to connect by using the engine name.

The default engine name is the first database name.

environment variable
The DOS and OS/2 operating systems allow users to set environment variables that can be used by applications to identify system-specific information.   Windows applications have access to DOS environment variables.   SQL Anywhere uses the SQLCONNECT and SQLANY environment variables to identify default connection parameters and home directory for SQL Anywhere.

entity
In Entity-Relationship design of databases, a first step is to identify the entities in the information you will incorporate into your database.   Entities become tables in the final implementation.

## Entity-Relationship design

Entity-Relationship design is a systematic approach to designing databases, based on a top-down analysis of the tasks you need to perform.

erase
Erasing a database deletes all tables and data from disk, including the transaction log that records alterations to the database.

ethernet
Ethernet is a term associated with network cards, a protocol, and a network topology.   In a network topology context, an ethernet is commonly associated with a bus network.

exception handler
In procedures and triggers, an exception handler is defined in the EXCEPTION part of a compound statement, and is code that is executed if an error is encountered in the compound statement.

extraction
In SQL Remote replication, the act of synchronizing a remote database with its consolidated database by unloading the appropriate structure and data from the consolidated database, then reloading it into the remote database. Extraction uses direct manipulation of ordinary files--it does not use the SQL Remote message system.

FILE
In SQL Remote replication, a message system that uses shared files for exchanging replication messages. This is useful for testing and for installations without an explicit message-transport system (such as MAPI).

For FILE, the addresses are subdirectory names relative to the directory defined by the SQLREMOTE environment variable. This is typically on a volume that is shared between the replicating databases. SQL Remote stores its messages as replication files in this subdirectory.

For example, if SQLREMOTE is set to S:\SQLANY50 (where S:\ is a shared volume), and a FILE address is set to SAMS, SQL Remote expects to find the replication files in the S:\SQLANY50\SAMS directory.

foreign key
Tables are related to each other by using foreign keys.   A foreign key in one table (the foreign table) contains a value corresponding to the primary key of another table (the primary table).   This relates the information in the foreign table to that in the primary table.

See also:   primary key, referential integrity.

forward log
See transaction log.

full backup
In a full backup, a copy is made of the entire database file itself, and optionally of the transaction log.   A full backup contains all the information in the database.   See also:   incremental backup.

function
Also called a "user-defined function", this is a type of procedure that returns a single value to the calling environment.   A function can be used, subject to permissions, in any place that a built-in non-aggregate function is used.

grant option
When a user is granted a permission WITH GRANT OPTION, they can grant the permission in turn to other users.

group
A user group is a database user ID that has been given the permission to have members.   User groups are used to make the assignment of database permissions simpler.   Rather than assign permissions to each user ID, a user ID is assigned to a particular group, and takes on the permissions assigned to that group.   See also:   permissions, DBA, user ID.

identifier
An identifier is any string composed of the characters A through Z, a through z, 0 through 9, underscore (_), at sign (@), number sign (#), or dollar sign ($).   The first character must be a letter.   Alternatively, any string of characters can be used as an identifier by enclosing it in double quotes.

incremental backup
An incremental backup is a copy of the transaction log.   This log contains all the information needed to restore the database to its present state from its state when the transaction log was started.

index
An index on one or more columns of a database table allows fast lookup of the information in these columns, and so can greatly speed up database queries.   Specifically, indexes assist WHERE clauses in SELECT statements.

inner join
An inner join is one kind of JOIN. operation allowed in the FROM clause of SELECT queries. INNER JOIN is the default type of join.

An inner join includes only those rows of the table on each side of the expression that has matching rows in the other table.

See also: outer join.

integrity

Integrity of information in a database ensures that each row in the database can be uniquely identified (entity integrity) and that relations between rows in different tables are properly maintained (referential integrity). SQL Anywhere provides several tools for maintaining the integrity of information in databases. See also: entity integrity, referential integrity.

IPX
IPX is a network-level protocol by Novell.

isolation levels
The isolation level for instructions within a transaction determines the extent to which other transactions may share the data addressed by the transaction. The isolation level can be set by the user: different isolation levels are appropriate for different kinds of transaction. See also: locking, transaction.

ISQL
ISQL (Interactive SQL) is a SQL Anywhere database administration and browsing utility.

join
The JOIN clause in a SELECT query enables a single database query to obtain information from several related tables.

keys
Database tables may contain two types of key:   a primary key , and a foreign key .

LAN
See local area network.

local area network
A local area network (LAN) is a collection of networked computers characterized by two attributes:

- A diameter of not more than a few kilometres.
- Ownership by a single organization.

See also: protocol.

locking
SQL Anywhere places a lock on a row that is being addressed by a transaction. The lock prevents other transactions from having access to the row in a way that could make the data in the database or the data seen by users of client applications inconsistent, while allowing concurrent transactions. See also: concurrency, transaction.

log files
SQL Anywhere maintains a set of three log files to ensure that the data in the database is recoverable in the event of a system or media failure, and to assist database performance.   See also:   checkpoint log, transaction log, rollback log.

MAPI
Microsoft's Message Application Programming Interface, a message system used in several popular e-mail systems such as Microsoft Mail.

A typical MAPI address may take the form **jsmith**.

media failure
A media failure occurs when the information on a medium (typically a hard disk drive) becomes unusable. A media failure may occur as a result of damage to the file, the file system, or the actual device. SQL Anywhere includes tools for backup and recovery to minimize the effects of media failure. See also: system failure, backup.

Message Agent
In SQL Remote replication, a program (DBREMOTE) that sends and receives replication messages on behalf of a database.   A consolidated database's message agent typically runs continuously, receiving replication messages continuously and sending replication messages periodically, while a remote database's agent typically runs on demand, receiving and sending all pending messages.

message system
In SQL Remote replication, a protocol for exchanging messages between the consolidated database and a remote database.   SQL Anywhere includes support for several message systems, each encapsulated in a message type.

message type
In SQL Remote replication, a database object that specifies how remote users communicate with the publisher of a consolidated database. A consolidated database may have several message types defined for it; this allows different remote users to communicate with it using different message systems.

SQL Remote includes the following pre-defined message types: FILE, MAPI, SMTP, and VIM.

These message types are automatically added to databases created in or upgraded to SQL Anywhere. To use these pre-defined types, you must define publisher addresses for them.

messages
Message based communication between applications or computers does not require a direct connection.   Instead, a message sent at one time by an application can be received at another time by another application at a later time.

named connection
Any connection from ISQL or and embedded SQL application to a database may optionally be given a name. If a client application has several connections in place simultaneously, the user may switch from one connection to another by using the SET CONNECTION command. See also: connection.

named pipes
Named Pipes are an interprocess communication mechanism implemented by a number of leading operating system vendors.   Named Pipes are usually implemented atop some transport-level protocol.

NDIS
NDIS (Network Driver Interface Specification) is a datalink-level protocol jointly defined by Microsoft and IBM.

NetBIOS
NetBIOS is a transport-level interface defined by IBM.   See also:   protocol.

NetBEUI
NetBEUI is a transport-level protocol.   See also:   protocol.

NetWare
A widely-used network operating system by Novell.   NetWare generally employs the IPX protocol, although the TCP/IP protocol may also be used.

network adapter
A network adapter is the physical attachment to a computer allowing network communication.   Also called a network card.

See also:   network driver

network architecture
A network architecture is the physical interconnection of computers on a network.   Typical architectures include bus, where all computers connect to a single carrier, and ring, where one computer is directly connected to another to form a closed ring.

network card
A network card is the physical attachment to a computer allowing network communication.   Also called a network adapter.

See also:   network driver

network driver
A network driver is the software logically located between the operating system and the network card , which allows applications to communicate to the network card.   Network cards are usually bundled with a number of network drivers for various operating systems and network protocols.

network operating system
An operating system optimized for server deployment, such as Novell NetWare and Microsoft Windows NT Server Edition.

network server
A SQL Anywhere database engine that runs on a different PC from the client application (which uses the SQL Anywhere client). The server communicates with the client using a particular network protocol.

A network server can support many users connecting from many PCs on the network.

NULL
The NULL value is a special value for a database entry, different from any other valid value for any data type. The NULL value represents missing or inapplicable information. See also: column, constraint.

nullability
Determines whether a column allows a NULL value to be assigned to it.   Columns are typically allowed to be NULL if their values are optional or not always available, and are not required for the data in the database be correct.

ODBC
The Open Database Connectivity (ODBC) interface, defined by Microsoft Corporation, is a standard interface to database management systems in the Windows and Windows NT environments.   ODBC is one of several interfaces supported by SQL Anywhere.

ODI
ODI (Open Datalink Interface) is a data link-level interface defined by Novell.

optimizer
The SQL Anywhere database engine contains an optimizer which attempts to pick the best strategy for executing each query. The optimizer uses educated guesses about the occurrence of particular elements in the database to select a strategy. The user can provide explicit estimates in order to help tune an execution strategy. See also: index, query.

outer join
An outer join is one kind of JOIN operation allowed in the FROM clause of SELECT queries.

An outer join may be either a LEFT OUTER or a RIGHT OUTER join.   An outer join includes all rows of the table on the LEFT (RIGHT) of the expression whether or not there are matching rows in the other table.

See also inner join.

owner
Each object of a database is owned by the user ID that created it.   The owner of a database object has rights to do anything with that object.   See also:   DBA.

packet
A packet is a communication entity between processes.   Messages between two processes are usually fragmented into a number of packets for data transmission from the source and then reassembled at the target host into a single message.

page
A SQL Anywhere database file is composed of pages.   Before a page is updated (made dirty), a copy of the original is always made in memory.   The copied pages are the checkpoint log.   The page-size of a database file is specified when the database is created.   See also:   cache, checkpoint log.

page size
The size of each database page (in bytes).   The page size can be 512, 1024, 2048 or 4096 bytes, with 1024 being the default.   Other values for the size will be changed to the next larger size.   Large databases usually benefit from a larger page size.

passthrough
In SQL Remote replication, a mode by which the publisher of the consolidated database can directly change remote databases with SQL statements. Passthrough is set up for specific remote users (you can specify all remote users, individual users, or those users who subscribe to given publications). In normal passthrough mode, all database changes made at the consolidated database are passed through to the selected remote databases. In "passthrough only" mode, the changes are made at the remote databases, but not at the consolidated database.

password
Whenever a user connects to a database, a password must be specified. The passwords are stored in the SYS.SYSUSERPERM system table, to which only the DBA has access.

performance statistics
Values that reflect the performance of the database system with respect to disk and memory usage. The CURRREAD statistic, for example, represents the number of file reads issued by the engine which have not yet completed.

permissions
Each user has a set of permissions that govern the actions they may take while connected to a database. Permissions are assigned by the DBA or by the owner of a particular database object. See also: DBA, database object, owner.

phantom lock
A phantom lock is one of three types of lock supported by SQL Anywhere to support concurrency, as part of SQL Anywhere's transaction processing capabilities.

A phantom lock is a type of read lock employed at isolation level 3.   No other transaction can read or update the row once a phantom lock has been acquired.

See also:   read lock, write lock, locking.

primary copy
In a replication installation, the primary copy of any data is considered to hold the original of the data. All other places in which the piece of data is held are copies of this original.

protocol
The rules and conventions used to communicate between computers are collectively known as a protocol. Instances of protocols include IPX, NetBEUI, and IP. The SQL Anywhere client and server can communicate using a number of different protocols, including NetBIOS, IPX, NamedPipes and TCP/IP.

See also: IPX, NetBIOS, Named Pipes, TCP/IP.

protocol stack
Network communications between communications take place according to a set of protocols.   These protocols are logically organized in layers, each with a different function.   The set of layers forms a protocol stack.

primary key
Each table in a relational database must be assigned a primary key. The primary key is a column, or set of columns, whose values uniquely identify every row in the table. See also: entity integrity.

programming interface
Programs may connect to SQL Anywhere using one of the interfaces supported by SQL Anywhere. Embedded SQL is SQL Anywhere's native interface. ODBC is another low-level interface. WSQL DDE and WSQL HLI provide higher level interfaces.

publication
In SQL Remote replication, a database object that describes data to be replicated.   A publication consists of articles (tables or subsets of tables).   Periodically, the changes made to each publication in a database are replicated to all subscribers to that publication as publication updates.

publication update
In SQL Remote replication, a periodic batch of changes made to one or more publications in one database.   A publication update is sent as part of a replication message to the remote database(s).

publisher
In SQL Remote replication, the single user in a database that can exchange replication messages with other replicating databases.

remote database
In SQL Remote replication, a database that exchanges replication messages with a consolidated database. Remote databases may contain all or some of the data in the consolidated database.

remote permission
In SQL Remote replication, the permission to exchange replication messages with the publishing database. Granting remote permissions to a SQL Anywhere user make them a remote user. This requires you to specify a message type, an appropriate remote address, and a replication frequency. In general terms, remote permissions can also refer to any user involved in SQL Remote replication (for example, the consolidated publisher and remote publisher).

qualifier
The name of a database object, such as a table or a view, may be preceded by a qualifier to indicate its owner and hence uniquely identify the database object.   For example, a table named **some_table** created by a user **A_User** can be uniquely identified as **A_User.some_table**.

Columns in tables or views may likewise be qualified by the table name (and owner) to identify them uniquely.   For example, a column **this_column** may have the fully qualified name **User.some_table.this_column**.

Use of fully qualified names in SQL queries helps to avoid ambiguities.

query
Information is retrieved from SQL Anywhere databases by submitting a query. A query is an SQL SELECT statement, the clauses of which specify the exact information the database engine must return tot he client application. See also: SQL, subquery.

read lock
A read lock is one of three types of lock supported by SQL Anywhere to support concurrency, as part of SQL Anywhere's transaction processing capabilities.

When a transaction reads a row, employing one of the more secure isolation levels available (level 2 or 3), other transactions can read the row, but cannot update it.

See also: phantom lock, write lock, locking.

referential integrity
The tables of a relational database are related to each other by foreign keys. SQL Anywhere provides tools that maintain the referential integrity of the database: that is, ensure that the relations between the rows in different tables remain valid. See also: foreign key, entity integrity.

relationship
A step in the Entity-Relationship design of databases, is to identify the relationships between the entities that you have identified.

One-to-many and one-to-one relationships become foreign key relationships in the final implementation, while a many-to-many relationship becomes a table.   See also:   entity.

remote DBA authority
The Message Agent should be run using a user ID with REMOTE DBA authority, to ensure that actions can be carried out, without creating security loopholes.

remote user
In SQL Remote replication, a SQL Anywhere user who has been granted remote permissions in a replication setup. When the remote database is extracted from the consolidated database, the remote user becomes the publisher of the remote database, able to exchange publication updates with the consolidated database.   While SQL Anywhere groups can also be granted remote permissions, note that users in these "remote groups" do not inherit remote permissions from their group.

replication
For databases, a process by which the changes to data in one database (including creation, updating, and deletion of records) are also applied to the corresponding records in other databases.   SQL Anywhere supports replication using SQL Remote or Sybase Replication Server.

replication frequency
In SQL Remote replication, a setting for each remote user that determines how often the publisher's message agent should send replication messages to that remote user.   The frequency can be specified as on-demand, every given interval, or at a certain time of day.

replication message
In SQL Remote replication, a discrete communication that is sent from a publishing database to a subscribing database.   Messages can contain a mixture of publication updates and passthrough statements (manual SQL statements such as DDL).

resource authority
Resource authority is the permission to create and modify objects of a database schema. Resource authority can be granted only by the DBA. See also: DBA, permissions.

role name
Each foreign key is assigned a name, called a role name, to distinguish it from other foreign keys in the same table. If no role name is specified by the user, the role name is set to the name of the primary table.

rollback
When a user sends a rollback SQL statement to the database engine, all work since the last savepoint or since the beginning of the current transaction is undone, and no changes are made to the database by any of the instructions.   See also:   commit, transaction, rollback log.

rollback log
A log kept in order to cancel changes made to database tables.   The rollback log is needed in the event of a ROLLBACK request or a system failure.   There is a separate rollback log for each transaction.   When a transaction is complete, its rollback log is deleted.

row
All data in relational databases such as SQL Anywhere is held in tables, composed of rows and columns.   Each row holds a separate occurrence of each column.   In a table of employee information, for example, each row contains information about a particular employee.   See also:   table, column.

row-level trigger
A trigger that executes BEFORE or AFTER each row modified by the triggering insert, update, or delete operation is changed.

runtime database engine
The SQL Anywhere Desktop Runtime engine is a royalty-free redistributable database engine.   The runtime database engine supports all the database manipulation language features of the full SQL Anywhere, with the exception of procedures and triggers.   Also, the runtime database engine does not employ a transaction log.

savepoint
Within a transaction, a SAVEPOINT statement allows flexibility in committing and rolling back work.   Work since the most recent savepoint can be undone using ROLLBACK TO SAVEPOINT.   The RELEASE SAVEPOINT disallows any future rollbacks to the most recent savepoint.   Before SQL Anywhere version 4.0, savepoints were called subtransactions.   See also:   transaction, commit, rollback.

schema
The structure of a database is called the schema.   The schema is held in the system tables.

The schema includes the complete definitions of each database object in the database, including tables, indexes, views, procedures, and triggers.

search condition
In SQL, a search condition is a WHERE clause in a SELECT statement.

session
Communications across a network may take place in a session, (also called a connection, or virtual circuit) or in a connectionless manner, using a datagram method.   When a connection is established, a route from the source computer to the destination computer is part of the session setup, and routing decisions are not required for every packet.

Network server editions of SQL Anywhere use sessions for their NetBIOS and local Named Pipes communication links.

SMTP
Simple Mail Transfer Protocol.   This is a message system supported by SQL Remote.   A typical SMTP mail address takes the form **boss@myown.com**.

SQL

Structured Query Language (SQL) is the language used to communicate to SQL Anywhere databases.   SQL is very widely used in database applications, and in order to ensure compatibility among databases, SQL is the subject of standards set by several standards bodies.

SQLCA
A SQL Connection Area (SQLCA) is a segment of database client application code that manages the connection parameters governing the connections between the client application and a database.   See also: programming interface.

SQL Remote
An asynchronous message-based replication system for two-way server-to-laptop, server-to-desktop, and server-to-server replication between SQL Anywhere databases.

server
In SQL Anywhere, servers are database engines--the programs that manage the physical structure of the database and process queries on its data.   Servers can be standalone engines or network servers.   In SQL Central, standalone engines and network servers are both called servers.

stand-alone engine
A SQL Anywhere database engine that runs on the same PC as the client application.   A stand-alone engine is typically for a single user on a single PC, but can support several concurrent connections from that user.

statement
SQL allows several kinds of statement. Some statements modify the data in a database (commands), others request information from the database (queries), and others modify the database schema itself. See also: SQL, query.

statement-level trigger
A trigger that executes after the entire triggering statement is completed.

store-and-forward
Store-and-forward exchange of information is typical of message based systems, and allows information to be exchanged without a direct connection between applications.

stored procedure
Stored procedures are procedures kept in the database itself, which can be called from client applications. Stored procedures provide a way of providing uniform access to important functions automatically, as the procedure is held in the database, not in each client application. See also: trigger.

subquery
A subquery is a component of a SQL query (SELECT statement) that is itself a query.   Subqueries are important tools in constructing complex queries to databases.   See also:   SQL.   query.

submission
In a SQL Remote installation, changes made to a remote database and sent to the consolidated database are a submission.   If and only if the changes are successfully applied at the consolidated database they will be replicated to other databases.

subscriber
In SQL Remote replication, a remote user who is subscribed to one or more of a database's publications.

subscribing
In a Replication Server or SQL Remote installation, a database that has subscribed to a replication or publication receives updates of changes to the data in that replication or publication.

subscription
In SQL Remote replication, a link between a publication and a remote user, allowing the user to exchange updates on that publication with the consolidated database.   The user's subscription may include an argument (value) for the publication's SUBSCRIBE BY parameter (if any).

subtransaction
See savepoint .

synchronization
In SQL Remote replication, the process by which SQL Remote deletes all existing rows from those tables of a remote database that form part of a publication, and copies the publication's entire contents from the consolidated database to the remote database.   Synchronization is performed during the initial extraction of the remote database from the consolidated database, and may also be necessary later if a remote database becomes corrupt or gets out of step with the consolidated database (and cannot be repaired using passthrough mode). Synchronization can be accomplished by bulk extraction (the recommended method), by manually loading from files, or by sending synchronization messages through the message system.

system failure
A system failure occurs when a power failure or some other failure causes a computer to fail while there are partially completed transactions.   See also:   media failure, transaction, backup.

system catalog
The system catalog is the list of all tables and views in the database.

The SQL Anywhere system view SYS.SYSCATALOG presents this information.

system object
In a database, a table, view, stored procedure, or user-defined data type that is pre-defined by SQL Anywhere. System tables store information about the database itself, while system views, procedures, and user-defined data types largely support Sybase Transact-SQL compatibility.

system tables
Every SQL Anywhere database includes a set of tables called the system tables, which hold information about the database structure itself:   descriptions of the tables, users and their permissions, and so on.

The system tables are created and maintained automatically by the database engine.   They are owned by the special user ID **SYS**, and cannot be modified by database users.

system views
Every SQL Anywhere database includes a set of views, which present the information held in the system tables in a more easily understood format.

table
All data in relational databases is stored in tables. Each table consists of rows and columns. Each column carries a particular kind of information (a phone number, a name, and so on), while each row specifies a particular entry. Each row in a relational database table must be uniquely identifiable by a primary key. See also: database, row, column, primary key, database object, owner.

TCP/IP
Transmission Control Protocol/Internet Protocol (TCP/IP) is a network protocol supported by SQL Anywhere See also: protocol, IPX, NetBIOS.

temporary table
Data in a temporary table is held for a single connection only. Global temporary table definitions (but not data) are kept in the database until dropped. Local temporary table definitions and data exist for the duration of a single connection only.

token ring
A token ring is a specification of a particular network architecture and protocol.   A ring architecture is a topology is which every computer on the ring has a wire in and a wire out and forms a closed loop.   A token is passed around the ring to arbitrate between computers that wish to communicate at the same time.

topics
Each separate panel in an online help system is a topic.

Transact-SQL
The SQL dialect used in Sybase SQL Server.   SQL Anywhere supports a large subset of Transact-SQL.

transaction
A transaction is a logical unit of work that should be processed in its entirety by the database (though not necessarily at once) or not at all. SQL Anywhere supports transaction processing, with locking features built in to allow concurrent transactions to access the database without corrupting the data. Transactions begin following a COMMIT or ROLLBACK statement and end either with a COMMIT statement, which makes all the changes to the database required by the transaction permanent, or a ROLLBACK statement, which undoes all the changes made by the transaction. See also: locking, concurrency.

transaction blocking
A transaction becomes blocked when it must wait for another transaction to finish before it can carry out its task. Proper design of transactions by application developers can minimize the occurrence of transaction blocks, which slow down database operation.   See also:   transaction, locking.

transaction log
A log storing all changes made to a database, in the order in which they are made. In the event of a media failure on a database file, the transaction log is essential for database recovery. The transaction log should therefore be kept on a different device from the database files for optimal security.

The SQL Anywhere Desktop Runtime System does not employ a transaction log.

transaction log mirror
An identical copy of the transaction log file, maintained at the same time. Every time a database change is written to the transaction log file, it is also written to the transaction log mirror file. A mirror file should be kept on a separate device from the transaction log, so that if either device fails, the other copy of the log keeps the data safe for recovery.

transaction processing
A database engine that supports transaction processing is capable of ensuring that the results of each transaction are either stored in the database in their entirety or not at all. Transaction processing is a key element in promoting secure and reliable databases.

translation driver
A translation driver is a part of the data link layer of a protocol stack.   Data link layers conform to either ODI or NDIS specifications, and translation drivers supplied by networking software vendors enable multiple protocol stacks, requiring both ODI and NDIS drivers, to operate with a single network adapter.

trigger
A trigger is a procedure stored in the database that is executed automatically by the database engine whenever a particular action occurs, such as a row being updated. Triggers are used to enforce complex forms of referential integrity, or to log activity on database tables. See also: integrity, stored procedure.

two-phase commit
A mechanism to coordinate transactions across multiple database servers; only needed for distributed databases.
See also: commit, transaction.

unload
Unloading a database dumps the structure and/or data of the database to text files (command files for the structure, ASCII comma-delimited files for the data). This may be useful for creating extractions, creating a backup of your database, or building new copies of your database with the same or slightly modified structure. You can also unload the data (but not the structure) of a particular table.

updates
In replication, each set of changes sent from one database to another is an update to a publication or replication.

upgrade
A major release of SQL Anywhere (formerly Watcom SQL) generally means a revised internal database format (to support new database features).   When you upgrade your SQL Anywhere software to a major revision (for example, moving from 4.0 to 5.0), you should upgrade your existing databases to the new database format (after making backups of them).

user-defined data type
A named combination of base data type, default value, check condition, and nullability.   Defining similar columns using the same user-defined data type encourages consistency throughout the database.

user account
Every connection with a database requires a user account.   The permissions that a user has are tied to their user account.   A user account consists of a user ID and password.

user ID
A string of characters that identifies the user when connecting to a particular database. The user ID, together with a password, constitute a user account. See also: permissions, connection, DBA.

validate
When the information in a database, or a database table, is checked for integrity it is validated.   See also: entity integrity, referential integrity.

view
A view is a computed table. Every time a user uses a view of a particular table, or combination of tables, it is recomputed from the information stored in those tables. Views can be useful for security purposes, and to tailor the appearance of database information to make data access straightforward. As a permanent part of the database schema, a view is a database object. See also: table, database object.

VIM
Vendor-Independent Messaging, a message system used in cc:Mail and Lotus Notes.   A typical VIM address takes the form **jsmith**.

Watcom-SQL
The SQL dialect used in SQL Anywhere.   Watcom-SQL conforms closely to ANSI SQL.

Winsock
Winsock is a specification of the socket library with extensions.   The socket library is a collection of functions used to interface to a number of different transport-level protocols.   The socket library was initially specified by the University of California at Berkeley for the TCP/IP protocol.

write file
If a database is used with a write file, all changes made to the database do not modify the database itself, but instead are made to the write file.   Write files are useful in applications development, so the developer can have access to the database without interfering with it.   Also, write files are used in conjunction with compressed databases and other read-only databases.

write lock
A write lock is one of three types of lock supported by SQL Anywhere to support concurrency, as part of SQL Anywhere's transaction processing capabilities.

When a transaction updates or inserts a row, it acquires a write lock on the row.   No other transaction can acquire a lock on the same row.

See also:   read lock, phantom lock, locking.

WSQL DDE
A high level programming interface to SQL Anywhere for Windows applications.

WSQL HLI
A high level programming interface to SQL Anywhere.

sqlcode
A numeric error code identifying database errors.   Negative values are errors, while positive values are warnings. SQLCODE 0 indicates successful completion.

See also sqlstate.

sqlstate
A numeric error code identifying database errors.   SQL Anywhere provides both the ODBC SQLSTATE, returned to ODBC applications, and the ANSI SQLSTATE, which is returned in the SQLCA to Embedded SQL applications.

atomic
A set of operations is atomic if it cannot be broken down into smaller pieces.   Transactions are referred to as atomic because they must either be processed entirely or not at all.   The body of a procedure or trigger can be forced to be atomic by adding the keyword ATOMIC after the BEGIN keyword.

dirty
A database page is considered dirty when a change is made to it.   Before a page is made dirty, a copy of the original page is made and held as the checkpoint log.

SQL Communication Area
The SQL Communication Area (SQLCA) is a data structure used for passing data between a client application and the database engine.   Both ODBC and Embedded SQL use one or more SQLCA's to communicate with the database engine.

user groups
Managing permissions of many users is simplified if you assign users to user groups, and GRANT or REVOKE permissions from those groups rather than from individual users.

SQL Descriptor Area
See SQLDA.

SQLDA
The SQL Descriptor Area (SQLDA) is a data structure used for passing dynamic SQL statements to the database engine in Embedded SQL.

serializable
A set of concurrent operations is executed in a serializable manner if the net effect of the execution is identical to the result of executing each operation in turn, without any concurrency.

An isolation level of 3 is needed to guarantee the serializable execution of transactions.

conversation
A conversation between a client application and a server application is an organized exchange of data.   The term is commonly used in discussing DDE operations.

NT service
In the Windows NT operating system, applications set up as NT services can run even when the user ID starting them logs off the machine.

Running a SQL Anywhere database server as a service under NT allows databases to keep running while not tying up the machine on which they are running.

service name
In DDE, the service name is the top level identifier for a DDE conversation.

All conversations between a client application and the WSQL DDE Server must use the service name **WSQLDDE**.

topic name
In DDE conversations using the WSQL DDE interface, the topic name identifies the user and optionally which database the user wishes to query.

The topic name must be of the form **''Username,Password,Database''** or **''Username,Password''**.

item name
In DDE conversations using the WSQL DDE.   interface, the item name specifies the action to be performed by the WSQL DDE server.

wildconnects
In DDE, a wildconnect is an attempt to start a conversation without a service name or a topic name.

Wildconnects are not supported by the WSQL DDE server.

links
In DDE, a link is a method of exchanging data.   Any DDE conversation takes place through a DDE link.

The type of link supported by the WSQL DDE server is called a cold link.   Some other applications support links called a warm link and a hot link.

cold link
In a DDE cold link, the client must request data, and the server immediately supplies the data.

Cold links are also sometimes called **manual links**.

warm link
In a DDE warm link, the server notifies the client when data that the client is interested in has changed.   The client must then request the data if it requires the new data.   Warm links are also called

Warm links are also sometimes called **notify links**.   WSQL DDE Server does not support warm links.

hot link
In a DDE hot link, the server sends data of interest to the client as soon as the data changes.

Hot links are also sometimes called **automatic links**.   WSQL DDE Server does not support hot links.

userid
A correct userid parameter of a connect string is required for a user to connect to a database, and identifies the user to the database.   Each user's permissions are defined according to their userid.

Open Database Connectivity
See ODBC.

Dynamic Data Exchange
See DDE.

database application
See client application.

client application
In a database context, a client application is any application that communicates with a database engine.

client side
The client side of a client/server arrangement may consist of software and hardware.

The client-side software consists of a client application, and in a network setup also of DBCLIENT SQL Anywhere application, together with the host operating system and necessary network software.

The client-side hardware is the computer on which the client-side software is being run.

server side
The server side of a client/server arrangement may consist of both software and hardware, in addition to database files.

The server-side software consists of the SQL Anywhere database engine, and in a network setup also of the host operating system and necessary network software.

The server-side hardware is the computer on which the database engine is running, and the devices on which the database files are stored.

Structured Query Language
See SQL.

SQL Anywhere Client
The SQL Anywhere Client is the DBCLIENT application.

network request manager
The network request manager is a component of the SQL Anywhere network server which handles interaction with the SQL Anywhere Client.

root file
Unless you specifically create multiple files for your database using the CREATE DBSPACE statement, the database root file is the file your database is held in:   the database file.

If your database is held in multiple database files, the root file is the first file created, which holds the system tables.   When supplying the database file parameter in a connection string, you need to supply the root file name.

unique constraint
A unique constraint identifies one or more columns that uniquely identify each row in the table.   A table may have several unique constraints.

See also:   constraint

primary key constraint
A primary key constraint identifies one or more columns that uniquely identify each row in a table. Imposing a primary key constraint on a set of columns makes that set the primary key for the table. The primary key usually identifies the best identifier for a row.

See also: constraint.

foreign key constraint
A foreign restricts the values for a set of columns to match the values in a primary key or uniqueness constraint of another table.   For example, a foreign key constraint could be used to ensure that a customer number in an invoice table corresponds to a customer number in the customer table.   Imposing a foreign key constraint on a set of columns makes that set the foreign key in a foreign key relationship.

See also:   constraint.

check constraint
A check constraint allows specified conditions on a column or set of columns in a table to be verified.

See also: constraint.

unique index
A unique index on a table is an index for which it is ensured that no two rows have identical values in all columns in the index.

foreign table
A foreign table is the table containing the foreign key in a foreign key relationship.

See also: foreign key, primary table, referential integrity.

primary table
A primary table is the table containing the primary key in a foreign key relationship.

See also: primary key, foreign table, referential integrity.

bind variable
In Embedded SQL, a bind variable is a host variable used to pass values to the database engine.

host variable
In Embedded SQL, a host variable is a C variable which is identified to the SQL preprocessor.   Host variables can be used to send values to the database engine or receive values from the database engine.

See also:   bind variable.

correlation name
In a SQL SELECT query, a correlation name can be given to a table. The correlation name must then be used to refer to the table in the rest of the query.

A correlation name is a convenient way of reducing error and effort in developing long queries that refer to tables with complex names.

join type
A JOIN clause in a SELECT query is one of several join types.   The join types supported by SQL Anywhere are cross join, natural join, key join, inner join, and outer join.

key join
A key join is a join type used in SELECT queries that take information from more than one table.   The key join restricts results to rows where the foreign key value in one table is equal to the corresponding primary key value in the other table.

join condition
For any join type in a join clause of a SQL query, except for a cross join, a join condition can be specified.

current query
The current query is the SELECT or INPUT command that generated the information displayed in the ISQL data window.

projection
The result of a SELECT query with DISTINCT specified has all duplicate rows eliminated, compared to the result that would be obtained without the DISTINCT specifier.

The DISTINCT result is called the projection of the result.

string
In SQL, a string is any sequence of characters enclosed in apostrophes ('single quotes'). To represent an apostrophe inside a string, use a sequence of two apostrophes. To represent a new line character, use a backslash followed by an n (n). To represent a backslash character, use a sequence of two backslashes (\).

dynamic link library
See DLL.

interprocess communication
In operating systems that enable several applications to run at once, interprocess communication allows applications to work together.

For example, Named pipes is a method of interprocess communication supported by OS/2 and Windows NT. DDE is a method of interprocess communication supported by the Microsoft Windows operating environment and the Windows NT operating system.

data buffer
In DDE, a data buffer is one of two pieces of information sent from a client application to the WSQL DDE interface. The other is the item name.

poking
In DDE, poking is one of three main ways for a DDE client to exchange information with a DDE server.   Poking sends information from the client to the server.

executing
In DDE, executing is one of three main ways for a DDE client to exchange information with a DDE server.   Executing sends a command from the client to the server:   no data is returned by a DDE Execute transaction.

requesting
In DDE, requesting is one of three main ways for a DDE client to exchange information with a DDE server.   A request lets the DDE server know that the client wants to receive the results of a query.

user
See user ID.

nonexclusive lock
A lock is nonexclusive when other transactions can acquire a lock of similar type on a row.   The read lock is nonexclusive, while the write lock is exclusive.

exclusive lock
A lock is exclusive when other transactions cannot acquire a lock of similar type on a row.   The write lock is exclusive, while the read lock is nonexclusive.

place holder
In Embedded SQL, are included in the SQL Descriptor Area to indicate places where a host variable is to be accessed.

cost
SQL Anywhere has a query optimizer that picks a strategy for executing each query by estimating the cost of different approaches, and choosing the route with the lowest cost.   The cost of a query is the number of disk read and write operations required to carry out the query.

outer reference
A WHERE clause in a subquery can refer to columns in tables that do not form part of the subquery, but do form part of the main query. This kind of reference is called an outer reference. A subquery that contains an outer reference is called a correlated subquery.

correlated subquery
A subquery that contains an outer reference.

ODBC-enabled
An ODBC-enabled application is a client application that has an ODBC interface implemented.   SQL Anywhere conforms to ODBC 2.1, level 2.

Many application design systems, such as Powersoft PowerBuilder, are ODBC-enabled applications.

ODBC Administrator
The ODBC Administrator is a Microsoft program included with SQL Anywhere for setting up ODBC data sources.
See also: data source.

updated
A row in a database is updated if one or more of its values is altered.

deleted
A row in a database is deleted if it is removed from the database.

inserted
Inserting a row in a database table creates a new row.

optional
A foreign key is optional if it is allowed to contain the NULL value.

See also: mandatory

mandatory
A foreign key is mandatory if it is not allowed to contain the NULL value.

See also: optional

concurrent
Multi-user versions of SQL Anywhere support concurrent applications:   separate connections which may address the same data in the database, running at the same time.   SQL Anywhere provides transaction processing and automatic row-level locking to ensure that information remains consistent and that each concurrent application sees a consistent set of data.   See also:   transaction, locking.

deadlock
A deadlock occurs when two or more concurrent transactions become blocked in such a way that they cannot become unblocked.   In this situation one of the transactions must roll back its work in order for the other transaction(s) to continue.

handle
A handle is an integer that uniquely identifies an object.   ODBC makes use of handles to identify environments, connections, and statements.

client/server
A software architecture where one application (the client) obtains information from and sends information to another application (the server).

In a database context, the server is a database engine, and the client is a database client application.   The two applications often reside on different computers on a local area network.

## DDE transactions

In DDE, a transaction is an exchange of information between the DDE client and the DDE server.   poking, executing,, and requesting are the three types of DDE transaction.

domain
Every column is associated with a particular domain: the data type and range of values that constitute valid data for that column. See also: data type.

grantor
The user ID granting a permission to another user ID is the grantor. See also: grantee.

grantee
A user ID receiving a permission from another user ID is a grantee. See also: grantor.

procedures
See stored procedure.

interface
The boundary between two distinct entities is an interface.   In the context of a protocol stack, an interface between the different layers governs how information is passed down and up the stack.

server name
See engine name.

database server
A multi-user database engine is also called a database server.

columns
All data in relational databases such as SQL Anywhere is held in tables, composed of rows and columns. Each column holds a particular type of information. See also: table, row.

tables
All data in relational databases is stored in tables. Each table consists of rows and columns. Each column carries a particular kind of information (a phone number, a name, and so on), while each row specifies a particular entry. Each row in a relational database table must be uniquely identifiable by a primary key. See also: database, row, column, primary key, database object, owner.

rows
All data in relational databases such as SQL Anywhere is held in tables, composed of rows and columns. Each row holds a separate occurrence of each column. In a table of employee information, for example, each row contains information about a particular employee. See also: table, column.

indexes
An index on one or more columns of a database table allows fast lookup of the information in these columns, and so can greatly speed up database queries.   Specifically, indexes assist WHERE clauses in SELECT statements.

views
A view is a computed table. Every time a user uses a view of a particular table, or combination of tables, it is recomputed from the information stored in those tables. Views can be useful for security purposes, and to tailor the appearance of database information to make data access straightforward. As a permanent part of the database schema, a view is a database object. See also: table, database object.

triggers
A trigger is a procedure stored in the database that is executed automatically by the database engine whenever a particular action occurs, such as a row being updated. Triggers are used to enforce complex forms of referential integrity, or to log activity on database tables. See also: integrity, stored procedure.

stored procedures
Stored procedures are procedures kept in the database itself, which can be called from client applications. Stored procedures provide a way of providing uniform access to important functions automatically, as the procedure is held in the database, not in each client application. See also: trigger.

data type size
For certain data types, the number of characters that the data type can contain.

data type precision
For decimal and numeric data types, the total number of digits (including scale) allowed in the number.

data type scale
For decimal and numeric data types, the number of digits allowed after the decimal point.

ignore trailing blanks
By default, trailing blanks are ignored for comparison purposes, and Embedded SQL programs pad strings fetched into character arrays.   For example, the two strings 'Smith' and 'Smith ' would be treated as equal in a database created with trailing blanks ignored.

The database creation option to ignore trailing blanks is provided for compatibility with the ISO/ANSI SQL standard. The default is that blanks are significant for comparisons, which was the only behavior supported in releases up to Watcom SQL Version 3.0.

SUBSCRIBE BY
In a subscription to a publication, a SUBSCRIBE BY value is matched against the SUBSCRIBE BY column or expression for rows in the table.   The subscriber receives all rows for which the value of the column or expression is equal to the SUBSCRIBE BY value.

Keyword Search

Glossary