What is Advanced Query Tool?
Advanced Query Tool is a simple, fast and easy-to-use database query utility, designed for the application developer and DBA.

It has four major features:

- allows you to very quickly see all the tables in your system, their definition and content. Queries(*) against the tables can be easily developed and saved.
- for some databases, AQT will read the **system tables** to provide you with comprehensive information about the objects in your system. Not just tables and views but also triggers, procedures, packages, tablespaces, security rights. This feature is unique to AQT.
- provides a large number of useful tools for the DBA and developer. See Primary Features of Advanced Query Tool for a list of these.
- has a powerful GUI query builder for easily building complex queries, including multi-table joins.

(*) Note: the term query is used rather loosely to refer to any SQL statement. AQT provides many features for updating your tables too.

AQT can access any database that can be set up as an ODBC data source. This covers most databases on the market.

The component of the product that reads the system tables has to be individually customized for each database type. So far this has been done for DB2, Oracle, Sybase, SQL Server (see database types for a full list of these). Typically 50 system queries are supplied. Other databases will be added in the future.

Advanced Query Tool has been extensively proven on the road. It was developed over a 4-year period by a Database Administrator frustrated with the inadequacies of existing tools. It can handle large databases, can handle unusual data types (Blobs, binary fields) and has many workarounds for peculiarities of ODBC drivers (these are all problems which bedevil other ODBC-based products). AQT works.

Users have found that it gives them unprecedented visibility into their system; its simplicity and ease-of use a great time saver.

Primary Features Provided by Advanced Query Tool
Advanced Query Tool offers the following main features.

Simple Interface
In developing this product, a large amount of effort has been invested into providing a simple, fast and easy-to use interface. Many users find the product worthwhile just for this.

- provides a very simple interface for viewing the tables in your database (both table definition and table content).
- you can do this with minimal clicking, minimal scrolling, minimal typing and minimal hunting-through-menus. See useability concepts for more on this.
- given a certain job-function, we guarantee that you can do this faster with AQT than any other product we have seen.

In Addition

- it can run against large databases and large tables, including mainframes. AQT runs happily and fast against databases with thousands of tables and with millions of rows. Most other ODBC-based products are very slow in such environments.
- It can handle many data types and other situations which normal ODBC-based systems have trouble with:
  - CLOB and BLOB columns can be displayed
  - binary columns can be displayed in either ascii or hex mode.
  - it has facilities for displaying and updating large columns (hundreds of bytes long).
  - It has work-arounds for dealing with the many bugs / peculiarities of ODBC drivers we have come across while developing and using this system.

Queries

- allows you to run SQL statements against your database.
- has a powerful GUI query builder for helping with the building of Select statements. It can also build update, delete and insert statements.
- you can apply a search condition (filter) to a table, to easily display particular rows of a table without a knowledge of SQL. This builds a query that can be reused or saved.
- queries can contain substitutional parameters. The values of these are prompted for when the query is run.
- queries can be saved for later use.
- queries can be read from a file and run. This includes running SQL scripts containing hundreds of SQL statements.
- results from a query can be sent to a file (in a variety of formats).
- a number of options are provided for analysing a table. This is a powerful way of quickly gaining an understanding of the data held in a table without having to write SQL.
- long-running queries or updates can be cancelled, or be set to time-out.
- once you have displayed a table, you can drill-down and examine the rows in detail.
- Has comprehensive facilities for updating data, both individual rows and mass-updates.
- updates can run under Transactional Control – the changes are not made until Commit (or Rollback) is selected.
- Has a Safe Update Mode which prevents accidental updating / deleting of more rows than intended. If more than one row is updated or deleted the user is asked whether they wish this

proceed or not (this option can be switched off or configured).
- has a comprehensive facility for running Stored Procedures.

System Queries
For some databases (*) AQT reads the system tables (a.k.a. catalog, dictionary) to provide you with comprehensive information about the objects in your system (not just tables but also triggers, procedures, tablespaces, packages, security rights etc).

This provides the user with a large amount of information useful for understanding and managing the database. Typically, 50 system queries are supplied. This feature is unique to AQT. The system queries are held in config files, and can be changed or augmented if required. AQT provides a tool for assisting with this.

(*) So far this has been done for DB2, Oracle, Sybase, SQL Server, MySQL, SQLBase (see database types for a full list of these). Other databases will be added in the future (AQT still works with databases other than these - just the system queries are not available).

See system-table query feature for more on this.

For the DBA
Being developed by a DBA, AQT has many features that make the life of a DBA easier.
- has a compare facility for quickly comparing the definition of a table between databases. This is useful if your tables are defined in more than one database. This is fast; you can compare (say) 50 tables in a few minutes.
- has a facility for simplifying the management of the security of your tables and views.
- can show view definitions in formatted / interpreted mode. With this you can easy match a view-column with the source of that column.
- can run multiple SQL statements. This can be used for running a script file containing a number of SQL statements. Options to pause / amend / retry makes Advanced Query Tool one of the best ways of running a large SQL script. The script file can contain hundreds or even thousands of SQL statements.
- DDL for your tables and views can be generated.
- can generate text containing table and/or column names. This has a large number of uses; from generating program code, to generating utility statements for your tables. If you have a large number of tables in your system this feature can save you a large amount of time and typing.

What's New?

New to AQT 4.2.7

This release fixes a few bugs:

- Query Builder failed if there were more than 500 columns in the query tables.
- Auto-detect of table relationships did not work for SQL Server
- Procedure information for DB2/390 was incorrect when there were multiple procedures with the same name.
- Show Table Detail was truncated if there were more than 50 values in the display.
- The Run SQL window did not show columns information beyond the 100th column.
- Improved the handling of tables which have a decimal-point as part of the table name (this often happens with Text data sources).

In addition, there are a few useability enhancements:

- in the Run SQL window you can Open/Save a query with Ctrl-O and Ctrl-S
- in the Update Row window you can run your update with F5

New to AQT 4.2.6

This release fixes a couple of urgent bugs:

- DECIMAL and NUMBER columns were displayed incorrectly in some cases.
- We have improved the way AQT behaves with DECIMAL and NUMBER columns when a **Decimal Separator** other than fullstop is used. Click here for a description of how this works.
- We now allow an Unlimited number of rows to be displayed.

New to AQT 4.2.5

This release fixes a number of bugs:

- Write LOBs to Files only worked when Max Column Size was set to No Limit, and not at all for unregistered users. Oops! This has now been corrected.
- Load LOBs from external Files has been extended to also work for long character columns.
- Miscellaneous other fixes with **Write LOBS to Files** and **Load LOBS from external Files**. These features have now been tested on a number of database types.
- SYBASE decimal columns now have the lengths reporting correctly.
- Print Data now does not overlay data when a column contains linefeed characters.
- For DB2/UDB we have improved the way we display User and Group authorities. AQT will now display the correct results when there is a User and Group with the same name.
- The positions of a number of windows are saved and reestablised when the window is next opened (previously many windows opened centered in your screen and, due to a quirk in VB, re-centered themselves when they regained focus).
- Date / Timestamp values earlier than 100A.D. are now displayed correctly.

New to AQT 4.2.4

New features:

- BLOB/CLOB values can now be loaded from an external file. This provides a very easy mechanism for loading BLOBs into your database.
- BLOB/CLOB columns can be written to files; this allows the full LOB value to be retrieved from the database. Mechanisms are provided to display the LOB using an external application.

A few bugs were fixed:

- Timestamp columns sometimes had the decimal part displayed incorrectly
- error displaying a table with more than 50 date columns.
- Compare Tables treated TIMESTAMP,TIMESTMP and CHARACTER,CHAR as different types.

New to AQT 4.2.3

A few bugs were fixed:

- binary columns were incorrectly displayed in some cases
- Oracle binary columns more than 1000 bytes were incorrectly displayed as Null
- Null numeric values were incorrectly shown as blank

New to AQT 4.2.2

This release has a few miscellaneous bug fixes. In addition:

- The Update Row screen now has an option Get Constants (the same as Get Constants in the Run SQL Window)
- We have changed the behaviour of the Table Info and Table Display windows slightly. In the past, if you clicked on a large value a popup box showed you the full value of the cell. This often happened when you didn't want it to. Now you have to double-click a cell to display the full value.

New to AQT V4.2.1

This release has a number of miscellaneous bug fixes.

- **Add Table to Query** has had a number of bugs and quirks ironed out.
- The **GUI Query builder** has a number of miscellaneous minor bugs fixed
- Some users were having trouble using AQT with **SQL Server V6.5**. This is due to the way parameterised queries were being processed. We have resolved this problem by not using parameters in SQL Server 6.5 system-table queries.

New to AQT V4.2.0

With this release we have completely re-written the database-access layer of AQT. We have removed Visual Basic's RDO layer and rewritten it with native-ODBC calls. While this is largely an internal change, it allows us to correct a number of long-standing problems, and will allow us to offer some extra features in the future.

In particular:

- AQT nows displays large columns without problem. In the past there were a number of problems with these (which manifested themselves in a number of ways).
- Blobs and Clobs are now displayed (previously they were excluded from the table display).
- AQT now displays timestamps correctly. Previously they were being truncated at the second. DB2, Sybase, SQL Server and Informix users will now see the correct timestamp value.
- Oracle users can now use the Oracle ODBC Driver for Oracle (in the past AQT only worked well with the Microsoft ODBC Driver for Oracle).

Other features this release has:

- support for Informix has been added.
- the window for running Multiple SQL Statements has been rewritten. This now shows you your

SQL statements before you run the Script. You can exclude particular statements from being run. You can also change the text of your statements before they are run. This makes AQT a very powerful tool for running a large script.
- you can now specify a statement delimiter other than semicolon.
- we've added some features to make it easier to view large text columns. On the Table Display window there is an option "View As ..." which allows you to view the value of a column with another program (Notepad, Internet Explorer, Word). This is extremely useful if you are storing html, xml or Word documents in a table.
- we've finally bowed to popular demand and allowed up to 100,000 rows to be displayed. Just don't complain to us if your PC runs out of memory. In the future you will be able to display **all** of a large table by being able to display the table in a series of "chunks".
- for DB2 users, we allow you to specify your Isolation Level (Committed Read or Uncommitted Read).
- the Date and Time formats are now remembered between sessions.
- the GUI Query Builder can now handle up to 20 tables, 20 joins and 200 Where clauses.
- the Build Where Clause window of the GUI Query Builder has a new "OK and Add Another" which makes it a lot quicker to add multiple Where clauses.
- The Table Display window now uses a Text Box in the right grid for displaying large objects such as View Text and Procedure Text. This gives the ability to scroll through the text.
- The Query Timeout option has been removed as we have decided it is unnecessary.
- The Signon window has another option - **Prompt for Missing Info**. In most cases is a better option than **Prompted Signon**.
- **Run Stored Procedures** had a few bugs removed and a few other miscellaneous enhancements.

A number of changes have been made to the Run SQL Window:
- you now see a list of options when you **right-click** the SQL Text box
- you have an option to do a **syntax-check** of your SQL
- you can set the **font** used in the text box
- you can change the **case** of selected text
- you can **switch** to another table without having to exit the window
- the last 10 **tables** you have dealt with are remembered
- the last 10 **queries** you have dealt with are also remembered

Dates

We have reviewed the way we deal with dates. These are complicated as:
- people want the flexibility to view them in a particular format.
- most databases are very fussy about how dates are specified in insert/update statements.
- all databases have different date/time fields, and different rules about how they are handled.

To cope with this we are introducing the concepts of **Display Format** and **Update Format**. Display Format is how you wish to see the dates displayed. This is how you will normally see the dates. Update Format is how the date needs to be formatted when specified in an update/insert statement (or a Where clause). AQT will show dates in the Update Format when:
- you do a Get Values to get the values of a date column
- you go into Update (or Insert/Delete) Mode on a table. AQT will "convert" the dates from the Display format to the Update format.

See here for more info on dates.

Oracle TO_DATE Function

With Oracle, dates often have to be specified in conjunction with the **To_Date** function. In most places where dates need to be specified, AQT gives you the option to add the To_Date function to the date. This is done in:

- the **Run SQL** window, where you may need to specify a Date in a Where clause
- the Add Where clause window of the Gui Query Builder
- the Update function of the Row Detail window

Not Done in V4.2

We have done our first cut of support for Informix. In future releases we will improve this. What has not been done so far is:

- support for Informix-syntax **outer join** is only partially implemented
- column-list for **synonyms** hasn't been implemented yet
- the number of **system-table queries** will be increased. We'd appreciate some suggestions here on the type on information you would find most useful.
- handling of **timestamp** columns will be improved

This is, of course, assuming we have some Informix users out there. If so, please contact us to let us know (features in AQT are generally based on demand).

Plus:

- **Run Stored Procedures** for some versions of **Sybase** for some ODBC drivers can give a variety of problems. We are working on this.

New to AQT V4.1.3

The main feature in this release is a facility for running Stored Procedures. This has the following features:

- prompts for the procedures parameters.
- handles all types of parameters – Input, Output and Input/Output.
- handles the output of multiple result-sets
- saves the details of the last 100 procedures run. These can be recalled and rerun. This history is saved between AQT sessions.

This feature is extremely useful for those running stored procedures on a regular basis. Sybase and SQL Server users who use Stored Procedures for system administration, will find this particularly useful.

Other features in this release are:

- Support for DB2/400 has been added.
- Saved / Retrieve queries now remembers the last 10 directories used.
- The Row Display window has been amended. The way the scrolling is done has been changed slightly; the window can now cope with tables of up to 500 columns.
- The GUI Query Builder now has a Query Assistant that provides information about your query.
- In the Run SQL window you can rebuild the column list for the table that is in your SQL.

New to AQT V4.1.2

In this release we have put in several suggestions we have received from a number of users.

- The columns in the **Table Display** window now auto-expand to the size of the data. This makes

the window a lot easier to read, especially on large-resolution screens.
- A number of minor enhancements have been made to the **Run SQL** window – click here for details.
- Many of the windows now have the Database name as the first part of the window caption. The database name is thereby more visible on the task-bar icons (useful if you are running multiple AQTs against different databases).
- Setting and saving of the **font** used in the Table-Display window and for Printing has been improved.
- The license keys are now held in the LOCAL_MACHINE, rather than CURRENT_USER, part of the registry. This is useful in an NT environment where there might be multiple users wishing to use AQT on the one machine. All other settings are held in CURRENT_USER.

New to AQT V4
- The major feature is a GUI Query Builder. This is a major new component of AQT – about 9 months development work. Click on the above link to read about the features of this.
- The GUI Query Builder can reverse-engineer SQL – eg. take an existing SQL statement and display it as a GUI query. You can thereby use AQT to amend (and reformat) your existing SQL queries.
- The GUI Query Builder can also reverse-engineer view definitions. In the Table Info window you now have an option "View as Query" to show the view definition as a GUI query.

Other Features

Apart from the GUI Query Builder, the other new features for this release:
- the Query History is now saved (on disk) between AQT sessions. The number of entries in this is increased to 100.
- Elapsed time for a query is shown, and this is included in the Query History.
- You can save the results of a query (to disk). This is different from the Output to File; in the Display Table window you now have an option **Results->Save**. This saves the results-grid to disk; you can retrieve this at a later point in time (even if you are no longer connected to the database).
- If you have saved the query results, this is shown in the Query History; the query results can be re-displayed with a single click. This is extremely useful.
- In the Run SQL window, you can run a part of the query by selecting it before hitting Run / F5 (it will run the selected text, rather than everything in the query window).
- Compare Tables can now compare tables in different schemas in the same database.
- Support for SQLBase has been added.

What's Coming Up

AQT V4.3

The main feature of this release will be the ability to be connected to multiple databases at the same time (even if they are databases of different types). If you deal with many databases you will be able to access all of them from a single AQT session.

- the Table Information window will show information from the multiple databases in separate child-windows. You will be able to switch between these with a single click, or tile the windows to view information side-by-side.
- in the Run SQL window, you will be able to select which database you want to run your query against.

In addition, the way the behaviour of the Run SQL, Table Display and Query Builder windows will be improved:

- you will be able to display multiple tables/query-results at the same time
- you will be able to switch between the Run SQL and table/query display windows (eg. table/query display will no longer be modal).
- you will be able to directly run a GUI query without first returning to the Run SQL window

Simple Query Builder

To make it even easier to build queries, we are developing a "simple" query-builder which can build simple queries with a few clicks (this feature may or may not be delivered in V4.3 depending on workload).

Batch Mode

AQT will be able to run in batch mode – eg. from the command line without any user interaction. This will include the ability to schedule a query to run at a particular time (this feature may or may not be delivered in V4.3 depending on workload).

Time Frame

We plan to release the Beta of V4.3 at the end of March 2002, and the full release at the end of May.

AQT V5

This release will provide the ability to "do things" with your database objects – create tables, create indexes, drop tables etc.

As with the system-table query feature that AQT currently has, this feature will be driven through config files once the structure has been developed. This will allow a large number of actions to be implemented, and for you to add your own if required.

In addition this release will have:

- generation of full DDL (including indexes, foreign keys, triggers etc)
- Compare Tables will compare other attributes of tables (eg. indexes, keys, triggers)

The time frame for this is the end of 2002. We may deliver some of the functions as extra-

purchasable features (we haven't decided on this yet).

Types of Databases Advanced Query Tool Can Access

Advanced Query Tool can access any database that can be set up as an ODBC data source. This covers most databases on the market.

For all ODBC databases the majority of the function of AQT is available. In addition, for the following databases the system-table query feature of the product is available:

- Oracle V7 and V8
- DB2/UDB for Unixes & NT V2, V5, V6, V7, V8
- DB2/390 V4, V5, V6
- DB2/400
- Sybase Adaptive Server Anywhere
- Sybase Adaptive Server Enterprise V11, V12
- Informix
- Microsoft SQL Server V6.5, V7, V8
- MySQL (using the MyODBC ODBC driver)
- SQLBase

The system-table query feature can be easily be added to other databases in the future (this is controlled by config files so can be done with minimal code changes).

For more on the config files, see Database Types and Config Files.

Other databases which AQT has been used extensively against are:

- **MS Access**. Can be useful if you've been given an mdb but don't have Access on your machine. Or if there is some problem with your mdb and Access is having trouble opening it.
- **Excel** (*). Ditto as for MS Access. You might have been given an Excel file, but don't have Excel. In addition, AQT gives you good control of the selecting the columns and rows which are to appear in a report. This is an area where Excel is a little clumsy.
- **CSV** Files (*) (eg. files with "comma-separated variables"). Once an ODBC Driver has been set up for these, AQT gives you a very simple and easy way of seeing the content of CSVs, and running reports against them. In our experience, using AQT for this is a lot faster than importing the CSVs into Excel and reporting from there.

(*) to access either of these, you must first set up an ODBC source pointing to the Excel file, or (for CSVs) to the directory containing the CSV files.

Note that for Excel, all your worksheets appear under **Show->System Tables.**

Database Signon

When you start Advanced Query Tool you will be presented with the Signon window.

- A list of all your ODBC Data Sources will be displayed. Select the database you wish to sign on to.
- If your database is not in this list you will need to add it as an ODBC database. This is done through the ODBC icon within Control Panel. Alternatively you can hit **Datasources->Manage Datasources** to link into ODBC setup for your database.
- If you have added a new Data Source from the Control Panel, hit **Datasources->Refresh List** to update the Data Source list.
- Enter a userid and password for database then click **Signon**.

What Userid and Password?

Occasionally people are uncertain about what userid and password to use here.
This is a surprisingly non-trivial issue; it depends on what database you are accessing, where it is located and how it has been configured. As a generalization:

- for Oracle, Sybase and SQL Server, it is the userid and password with which you have defined to the database. The userid may be the same as your Unix/LAN userid, but the password will be one you've been set up in the database with.
- for DB2, it is the userid and password valid for the machine where the database resides (DB2 links into the security system of the operating system on which it is running). In other words use your normal 390 / Unix / NT userid and password.

If in doubt about which userid/password contact your friendly Database Administrator, or if you don't have one, any Database Administrator.

Client Security / Integrated Logon

Some databases do not require you to sign on, instead the database uses the security system of your PC. With this, your identity (eg. userid and password) is assumed to have been authenticated by your client (which may be NT) or LAN server. When this is in effect, a userid and password do not need to be supplied.

This is known as Client Security (DB2), Integrated Logon (Sybase), NT Authentication (MS SQL Server) or OS Authentication (Oracle). Whether these is used with your database will depend on your database type and how it has been configured. To find out whether this is used, contact your DBA.

- To use client security with Advanced Query Tool, leave the password field blank. AQT will sign on specifying neither userid nor password.
- If this succeeds you will be signed on to the database with the same userid you are signed onto your PC or LAN with (not the userid which you may be in the userid field on the Signon window).

Userid has No Password

If your userid has no password you can't just leave the password field blank; if you did AQT will assume you are doing an integrated logon and pass neither the userid or password in the signon request. In this case you need to click on **More Connect Options** and select **No Password**.

What ID are you to the database?

As a further complicator, for some databases the id you are known to the database as may be different to the id you signed on with (some databases map ids from external ids to internal ids). To see what id the database knows you as, see **Help-> Database Details->User Name** from the Table Information window.

Database Name

For SQL Server and Sybase Enterprise you can specify which database you want to sign on to. Click on "More Connect Options" and specify the database name. By default you sign on to the database specified in your profile.

Prompt for Missing Info

If you need to specify addition information for a connection, specify **More Connect Options** then select **Prompt for Missing Info.** This option is useful when, for instance, opening an MS Access database – you will be prompted for the mdb to open.

Show Password

If you are getting an error specifying your password, and you really want to see what password you have typed, click on **More Connect Options** then select **Show Password**. This will show your password in clear text.

Note: this option is disabled after you have signed on (to prevent anyone seeing your password). It can only be re-enabled once the password field is cleared.

Database Signon Problems

If you are having problems signing on, or you need to enter some database-specific parameters, you should do a Prompted Signon. Prompted Signon invokes the logon system supplied by the database vendor, giving you better control of the signon process.

System-Table Query Feature

For some databases, Advanced Query Tool reads the system tables to obtain the information about the objects in the system (the system tables are also called the dictionary or system catalog, depending on the terminology of you database). With such databases AQT can give you comprehensive information about your tables, and can report on objects other than tables (indexes, tablespaces etc).

See Types of Databases AQT Can Access for a list of what databases this feature is available for.

For other databases, AQT gets it information on system objects by using the ODBC API calls. These (generally) work fine; however the amount of information available is very limited – all you will see are tables, and within these you just see the table column list, primary key and (for some databases) the tables access rights.

Even for databases for which the system-table query feature is available, you can switch off this feature and run in ODBC table-info mode. You might do this if you are getting messages indicating a problem with the system queries. This setting is controlled by Options->General Options->Table Info.

The system-table query feature is controlled by config files. See Database Types and Config Files for a discussion on how this works.

Database Types and Config Files
The system-table feature of Advanced Query Tool reads the system tables (dictionary / catalog) for the database. These system tables are different for different database types (Oracle / DB2), different for database implementations (DB2/UDB for AIX is different from DB2 for OS/390), and even different between releases of a database.

To provide a simple and customizable way of reading the system tables, the SQL statements for all system-table queries are held in config files. There are no SQL statements hard-coded in the AQT code. The config files also gives the structure for database explorer, plus the various list-boxes and menus used for navigating through the system objects. By designing AQT in this way, the product has tremendous flexibility:

- if you want, you can amend these config files to add in your own system queries, or change the existing ones.
- you can distribute different config files to different categories of users.
- support for new database types can be implemented by building and distributing a new config file.

If you wish to amend any of the config files you should see Customizing your System.

The use of config files however provides an element of complexity as follows: when AQT connects to a database it needs to work out which config file to use. The discussion below describes how this works within AQT. You may need to know this if you are making changes to the config files, or if you are having problems with the system.

Database Type
AQT operates on the principal of auto-detection of the database type (Oracle, SQL server etc). When AQT connects to a database, it issues the GetInfo ODBC API call to get the database type. AQT is reliant on the database returning sensible information to this call and the type being known to AQT (see the next section). If you wish to know what name is being returned here, see **Help->Database Details->Database Type** or **Options->Config Files.**

Config File
Having found the database type, AQT then needs to determine which config file to use for this database. AQT uses a cross-reference between database type and the config file name. This cross-reference can be seen in **Options->Config Files**. The entries here are initially populated when AQT is run for the first time, however you can amend them if you are adding a new database type or if your database identifies itself by a name different to what AQT expects.
See Options – Config Files for a description on how these entries work.
Future releases of AQT will add extra entries into this list, as support for other databases is included. You may need to manually add an entry in this list if you download a new config file from the Cardett web site (www.cardett.co.nz).

Config File Directory
The directory where the config files are held is give by **Options->File Locations->Location of Config Files**. By default this is the same directory as the AQT executable. If your config files are held elsewhere you will have to change this.

See Location of Config files for more information on this setting.

Transaction / Commit Mode

By default, Advanced Query Tool operates in auto-commit mode. With this, a commit is done after every SQL statement has been run.
You can change this behaviour by entering **transaction mode**. With this, your changes are only committed to the database when you explicitly do a commit (or rollback).

These options are controlled by the **Transaction** menu option on the Run SQL window or Row window.

- selecting **Begin Trans** will put you into transaction mode.
- selecting **Commit Trans** will commit your database changes and take you out of transaction mode
- selecting **Rollback Trans** will roll back (undo) your database changes and take you out of transaction mode

If you are running SQL statements from a file, the commands COMMIT and ROLLBACK will also do a commit / rollback, and keep you in transaction mode (if this was selected).

While you are in transaction mode you can run any statements (including Selects to verify that your changes had the desired effect). You will get a reminder that you have some pending changes.

Please note that by using transaction mode you will be holding locks on rows until you commit. This can lock out other users. You should use this option with caution and never on a production system.

You will not be able to exit Advanced Query Tool until you make a decision to commit / rollback. If the program crashes or is killed then the changes will be rolled back.

Running Multiple Action Statements

In the Run SQL window you can run multiple SQL statements. Generally these would be action statements (rather than select statements).

This feature is often used to run a series of SQL statements supplied in a file OR a series of generated statements.

- the statements may be on the same or different lines (line breaks are ignored).
- the SQL statements are delimited by a semicolon.
- for Sybase and SQL Server, "go" at the start of a line is also taken as a statement terminator.
- for Oracle a / at the start of a line is taken as a statement terminator.
- any line starting with – (two minus signs) is a comment and is ignored.
- for Oracle any line starting with REM will be ignored.
- some databases allow comments to placed between /* and */. AQT currently does not support this, you could get "unpredictable" results if you have semicolons inside /* */ comments.

When you run a series of SQL statements, the **Run Multiple SQL Statements** window will be displayed. This will show you all the SQL statements in your script. Your statements are not run immediately, instead you must click on Run (or hit F5) to run them. This gives you the opportunity to:

- review your SQL statement before running them
- amend any statements if necessary
- prevent any particular statement from running (by clicking on the Run? column)
- start the script at a particular statement
- change some of the run options

Auto-Commit

By default, AQT will operate in Auto Commit mode. With this, every statement is committed after it has run. You can switch off Auto Commit mode; AQT will only commit when it hits a Commit / Rollback in your Script. Alternatively you can manually Commit / Rollback by clicking in the Commit / Rollback buttons.

Stop on Error

This option specifies whether AQT is to stop processing the script if there is an error processing an SQL statement. You have options:

- **Yes**. Stops on any Error
- **Yes, but ignore errors on Drops**. Stops on any error, except if it occurs on a Drop statement. The reason for this option is that scripts for building a database often consist of a number of Drop and Create statements. The Drops will fail if the objects do not already exist; this is an "expected" error and you may not want the script to stop running when this happens.
- **No**. Do not stop on any errors.

Other Buttons

- **Abort**. This cancels the currently-running statement. This will only be active if you have specified Async Queries (See Options->Statement Options).
- **Pause**. This stops the script after is has finished the currently-running query.

- **Skip**. If your script has stopped because of an error, you can either a) amend the statement then click on **Run** to re-run the statement or b) click on **Skip** to skip this statement and continue from the next statement.

Statements Delimter
By default, your SQL statement are delimited by a semicolon. If you want to change this, use **Options->General Options->Statement Delimiter?**

Displaying Table Information

After you sign onto the database, you will be presented with the Table Information window. This shows you information about the tables (and other objects) in your database.

What you see in this window will depend on the type of your database and how AQT has been customized ….. the following description may or may not be applicable to what you see.

Most of these options and features will only be available if the system-table query feature is enabled.

On the window you will see :

- the **database explorer** on the left. This shows the different types of objects in your database (Tables, Views, Indexes, Triggers etc).
- a **middle grid**. This shows all the objects (eg. tables) for a selected object-type.
- a **right grid**. This shows the properties of the object that has been selected in the middle grid.
- the panes between the grids can be adjusted. The screen can be resized. These setting are retained from one AQT session to the next.
- the data in the grids will automatically adjust their width to the size of the data. You can set this off (eg. to have fixed column-widths) by clicking on the **Auto-expand columns** icon.

Using the Database Explorer

The Database Explorer shows the different types of objects in your database (Tables, View etc), each represented by a particular icon.

- When you click on an object icon (such as Tables), the high-level grouping of the object is populated. This high-level grouping is called the Owner, Creator or Schema (depending on the terminology of your database). These can be thought of as being folders used to group and organize the objects within that object-type (however unlike files, the objects can't be moved from one folder to another).
- clicking on one of the Schemas will show you (in the middle grid) all the objects (eg. tables) within that Schema.
- some object-types do not have a schema associated with them. Tablespaces and Users are examples of these. When you click on the object icon in the Database Explorer the middle grid is immediately populated. The same comment applies to some "simple" databases (such as Access), for which tables do not have a schema.
- the Database Explorer can be hidden, to maximize valuable screen real-estate for the two grids. To hide it, click on the x at the top-right of the Database Explorer. To reestablish it, click on the **Database Explorer icon** in the main tool-bar.
- the **Refresh List icon** in the Database Explorer toolbar is used to reload the information in the Database Explorer from the database. You might do this if, for instance, you have created a new Schema.
- if the Database Explorer only shows you "Tables and Views" and "System Tables" then either you are using a database which doesn't have system tables (eg. Access) or you do not have the system-table query feature enabled.

Using the Middle Grid

The middle grid shows all the objects within the schema selected in the Database Explorer.

- clicking on an object in the middle grid will show you (in the right grid) the information for the object.
- generally there are a number of things that can be displayed about an object. For instance for tables you can display the columns in the table, the access list for the table, the views that are based on the table, the content of the table plus much more. You can see all these options by either right-clicking on the middle grid, or clicking on the list-box at the top of the right-grid. Selecting one of these options will display that information in the right grid.
- What options you see when you right-click the middle grid will depend on what system-queries have been set up for the database. These vary from one database-type to the next, as each database stores different information in the system tables. If you wish, you can change these queries or set up new ones. See Advanced Customisation.

0 Other notes on the middle grid

- There may be more than one option for what information is displayed in the middle grid. When this is the case a listbox with the options will appear at the top of the grid.
- Some queries may not work if either you do not have permission to read the system table, or if you are using an old version of the database that is not compatible with the AQT queries.
- If you want to quickly move through all the objects in the middle grid you can use the up and down arrows. This is a lot quicker than clicking on the individual objects.
- If you have a large number of items in the middle grid, you can click on the **Find** icon to specify a Filter to restrict the objects displayed.
- In the Evaluation version of the product, a maximum of 1000 objects will be shown in the middle grid.

Using the Right Grid

Sometimes, what is displayed in the right-grid are in themselves database objects (indexes, tables etc). This can be easily seen as the icons will be displayed with the objects. In this case you can right-click on an object to drill-down and display the properties of that object.

Other Notes

- **Sorting**. You can sort the contents of a grid by clicking on one of the **Sort** icons. This will sort the grid by the last column you clicked on.
- **Copying**. You can copy the contents of a number of cells by selecting the cells then hitting Ctrl-C or the **Copy** icon. This is useful to (for instance) copy the names of all your tables (to the clipboard). You can do more advanced copying with the Text Generator feature.
- **Database** For MS SQL Server, Sybase Enterprise and MySQL there will be an addition listbox **Database** appearing above the Database Explorer. This shows you all the databases within the current server – you can select one of these to switch into that database.
- **View Definition (formatted)** and **Table Access Grid**. These two displays require further explanation. Click on the links for further info.

Other Menu Items

- Display Table will display the contents of the table
- **Run SQL** takes you to the Run SQL window where you can analyze a table and run queries against it.
- **Build Query** takes you the Gui Query Builder where you can build a query against your table.

- Tools->Compare allows you to compare the definition of your table between this and another database.
- **Tools->Security Helper** takes you into the Security Helper window
- **Tools->Generate DDL** takes you into the DDL Generator
- **Tools->Generate Text** takes you into the Text Generator
- **Tools->Run Procedure** takes you into the Run Procedure window

View Definitions

When you display the information for a view you will see that there are two options: **View Definition** and **View Definition (formatted)**. I explain here what these are.

In most systems, the definition of a view is held as free-format SQL text. The menu option **View Definition** will show you this SQL Text (for some systems the text is held as a number of lines; these lines are combined for this display).

However often this display isn't very illuminating. If you are trying to understand how a view is defined you really want to know how the individual view columns are defined. View Definition (formatted) is an attempt to provide you with this information.

When you select View Definition (formatted), AQT will get the view SQL text then try and interpret it. The idea here is to decompose the SQL text into the individual columns then match this up with the view columns.

The result is presented in a two-column grid with the view-column (on the left) and the source of that column beside it on the right.

In addition the WHERE clause is broken into components and displayed over several lines.

To demonstrate the use of this feature, run it against some of the system views.

This function works well for most of the views we have run it against. However there are limits to what it can do; if you have a particularly complex view then this display could fail or look quite strange.

Table Access Grid
This section is relevant for Oracle, Sybase Enterprise and MS SQL Server users.
In the Table Information window, when you display the information for a user you will see an option **Table Access Grid**.

The **table access grid** runs a special program in AQT to display the user's access rights in a easy-to-read format. This format has one row per table the user has access to, and one column for each table privilege. The column privilege will be either N (does not have this privilege), Y (has this privilege), G (has this privilege with the Grant option).

Table Authorities will show you the access rights in the "normal format" (eg. in the way this information is held within the catalog). You can flip between these two displays if you want to compare how they present the same information (note however that Table Authorities also shows Grantor; this is not shown in the Table Access Grid).

By using the Table Access Grid, the table authorities for all databases types are displayed in an identical format. Partly one reason we have developed this display is to allow the Security Helper window to operate consistently across database types.

Displaying a Table

There are three ways of displaying the content of a table:

- in the Table Information window, click on the **Display Table** menu option / icon.
- in the Table Information window, click on the table, right-click then select **Show Contents**. This will display a "sample" of the table (first 30 rows) in the right-hand grid. Tables will be displayed as you select them by clicking on them (or using the up and down keys to scroll through the tables).
- in the Table Information window, click on the **Run SQL** menu option / icon to bring up the Run SQL window. To display the table, hit enter or click on Run. You can use this window to build more complex queries on the table.

Once you do any of these you will be taken to the Table Display window.

Notes:

- not all the rows in the table will be displayed. By default only the first 50 will be shown. This default can be changed by the Options menu option. If you are running the Evaluation version of the product you will not be able to increase this beyond 50.
- for very long columns, by default only the first 200 bytes of the column will displayed. This default can be changed by the Options menu option.

Table Display Window

This shows you either a table, or the result of a query. There are number of things you can do with this display.

View As

New to V4.2 is improved ability for dealing with large columns. When you right-click a cell you have options **View** and **View As**.

- **View** will show you the column in a larger text box. (the same one you get when you just click on it).
- **View As** will "send" the column value to another program. This is useful if the column contains data which is used by a particular application. Examples are: html, xml.
- By default, View As will have two options **text** and **html**. Selecting one of these will send the column value to the default application for those file types.
- To add another type to the **View As** menu, select **Add Another**. Specify this in the form title,type – example: xml,xml. Title is the name by which you want this type known. Type is the file type. AQT will create a file of this type and will "Open" this file. Which application gets used to open this depends on the Windows default application for the file-type.
- At this stage we have not supplied a means for changing or deleting entries from the View As list. This will be added in a later release.

Saving the Results

You can save the data in this display (to disk) and retrieve it later. This is very useful if you are displaying some useful data that you may wish to refer to at some later time.
Click on Results->Save and specify the name of the file where you want the results saved. By default this will be give a timestamp-type name in the Saved Queries directory. This is saved in a tab-delimited format that can be imported into Excel if you wish.
You can use Results->Retrieve to redisplay a saved results-grid.
When you have saved the results of a query, this will be shown in the Query History window - a little table-icon will appear in the Saved column. You can redisplay the results by clicking on this table-icon.

Compare to Saved

New to V4.1.3 is the option **Compare to Saved**. This provides a simple method of comparing the contents of two tables. Or comparing the same table at two different points in time.
Compare to Saved will compare the existing Results Grid to a Results grid you have saved (using **Results->Save**).
To compare two tables:

- display one of the table and go **Results->Save**
- display the second table and go **Results->Compare to Saved**. Open the file that you saved in the previous step.

It is recommended that when you display the tables you specify a **Order By** clause to ensure the rows are displayed in the same order.
When you click on OK, AQT will compare the tables;

- if any values are different you will be shown a window giving you details of the row/column which is different.
- you will have the option to "Bypass this Column". You can use this to exclude a column from the

Compare.
- once the Compare completes you will get a message at the bottom of the window giving the number of values compared successfully.

You can also use this feature to compare a table at two points of time. This is very useful if your table contains time-varying data, or you want to compare it before and after an update.

Finding Rows

If you wish to show only particular rows in your table, click on the Find icon. This will take you to the Find Rows window where you can apply a search condition to the table. You will then be shown only the rows that meet this search criterion.
This is a useful option if you are not familiar with coding searches in SQL.
Once you return to your SQL statement you will find that the Where clause has been added to your SQL. This serves as a very fast and easy way of adding Where clauses to your SQL.

Changing the Columns which are Displayed

You can change the order of the columns on this display, and remove columns from the display, by clicking on the **Layout** menu item, or by using one of the associated icons. Using these you can move columns left, right or delete them (from the display, not from the table!). Deleted columns can be restored. You can move/delete a range of columns by selecting the columns before doing the move/delete.
Once you return to your SQL statement you will find that your SQL has been amended to reflect the new column order. This provides a quick way of building a query using a particular set of columns.
This feature does not work in some rare circumstances: if your query is a multi-table join and one of your tables is a synonym, alias or resides in another database.

Showing Row Detail

To show the detail of a row, either a) click on **Row->Show Row Detail,** b) click on the Row Display icon c) right-click a row or d) hit F2. This will take you into the Row Display window. This window will also allow you to update or delete the row, or add new rows.
Note: in some previous releases of AQT you could also go into Row Detail by clicking on the left-most column in a row; this behaviour has been removed.

Showing Large Columns

If you have a large column you will notice that only the first part of it is displayed. To show the entire value of the column, click on the cell containing the column. How much of the column you see here depends on your setting for Options->Display Options->Max Column Size.

Printing Data

Clicking on the **Print** menu option will show you your two print options:
- **Print Screen** – this prints the screen as you see it
- **Print Data** – this is a more comprehensive print facility for the data. This is described in Printing Data. Note this option just prints the data shown in the grid.

Copying Data

To copy some cells (to the Windows clipboard), select the cells then hit Ctrl-C, click on **Edit->Copy,**

or the copy icon.

- as you select the cells, information on the range of cells you have selected will appear on the status bar at the bottom of the window.
- once you hit Ctrl-C / Copy you will be prompted for a number of copy options. These are discussed in more detail in Hints on Grid Usage.

Updating, Deleting, Inserting Rows

To do this, click on **Row-> Update mode**, Delete mode etc, or click on the appropriate icon. These will take you into Update Mode on the Row Display window.

Showing Column Information

You can see the attributes of the columns in your result-set by clicking on **View->Column Info**. Note that this shows you the attributes of your columns as reported by ODBC. This can be different from the columns attributes as shown in the Table Information window.
This display has a column called **source**; this shows you the clause in your SQL statement that is the source for the display column. This is useful when displaying the result of a complex query.

Sorting

You can sort the contents of the grid by clicking on one of the sort icons. This will sort the grid by the last column you clicked on. This is a "simple" sort and sometimes doesn't work correctly with some data types, such as timestamps.
Note: this only sort the data that is in the grid. It does not sort the entire table. If you want to sort the table, include an ORDER BY clause in your SQL statement.
Note: if you have deleted some columns from the display, after you've done a sort you will not be able to do "Undo Last Delete" (AQT will not be able to restore the column in the new order).

Selection Mode

You can specify what gets selected / highlighted when you click on the grid. This is known as Selection Mode – you can specify this within the **View** menu item. Free means only the click-on cells will be selected, Row means the entire row, Column means the entire column. Row and Column can be very useful for selecting an entire row/column for copying. Row mode is also useful for identifying a particular row when you are scrolling left and right.

Changing the Font

Use this if you wish to display (and print) your data using a different font. Click on the **Font** menu item then select the required font and font size.

- If you've increased the font size you may need to resize some of the column widths so that all the data fits in.
- Font styles (italics, bold etc) are not implemented.
- If you change the font, it will be remembered (even between AQT sessions).

Refreshing the Display

If you have run an update and changed the data, hit **Refresh** or **F5** to redo the query and redisplay the data.

SQL Window

Sometimes you will be shown the SQL icon. Clicking on this will take you to the Run SQL window. The SQL for displaying the table (including any search conditions) will be displayed.
Note: you do not get this icon if you entered the Table Display window from the Run SQL window. If you did, return to it by closing the Table Display window. All very complicated. Depends on the parent /child relationship of these windows.

Closing the window

Rather than closing the window by clicking on the x at the top right, it is recommended that you click on **Close**. This window is actually hidden rather than closed. You will be able to redisplay the window in the SQL window clicking on **Window->Results** or hitting Ctrl-R.

Row Display

The Row Display window is entered from the Table Display window. It is used for both displaying the details of a row, as well as providing an easy way of updating your data.

Displaying Rows

When you enter this window, the current row will be displayed.

- use the arrow icons or the **Goto** menu-item to move to other rows in your table.
- if you have a particularly large column value, click on the row then hit **F2** or **Column->Extended Edit** to see the row in a larger window.
- clicking on **Print** will allow you to print the data. There are two options for this; Print Screen and Print Data.

Updating Rows

Most of the other features of this window are to do with updating / deleting / inserting of rows in your table.

When you want to change your table, click on **Row** from which you can select one of the "update modes":

- use **Update** mode for updating this (or other) rows in the table.
- use **Delete** mode for delete this (or other) rows in the table.
- use **Insert Dup** mode for inserting a new row in the table using the values of the current row as a base.
- use **Insert New** mode for inserting a new row in the table with the columns initialised to default values (*).

(*) these will be the default values of the columns as defined to the database. If these aren't available, the default values will be blank, zero or current date/time depending on the data type.

When you select one of these modes, some new columns will appear on the window and some extra buttons will appear. These operate as follows:

- **Update mode**. When you select this you will get two extra columns Where and Update. You use Where to specify which **rows** are to be updated. You use Update to specify which **columns** are to be updated. If you just want to update one row, the Where columns is the primary key column, however any column(s) can be selected. This window can therefore be used to update either a single row in your table, or to update many rows in the table.
- To understand what the Where and Update options are doing, click on **Preview SQL** to see the SQL which is being generated.
- If you table has a Primary Key, by default these columns will be selected as the Where columns when you enter this mode. For Oracle and Informix users, if you have the ROWID in the result-set, this will be used as the Where column.
- **Delete mode**. When you select this you will get one extra column in the grid Where. You use Where to specify which columns form the Where clause for the delete (see Update Mode for more on this).
- **Insert**. When you select this you will get one extra column in the grid Include. You use this to specify which columns are to be included in the generated Insert statement. By default this is all columns, however you might wish to exclude some columns from this if you want them to pick up the default or generated value for the column.

Other Notes

- if you have a large text field, hit **F2** to enter **extended edit** for the value.
- you can get a drop-down list of all the existing values of the column. Do this by hitting **F3** or clicking on **Column->Get Values**.
- if you have changed your column values but want to return to their original values, click on **Reload**.
- after you have updated / deleting / inserted some data, these changes will not be reflected in the Table Display. You must hit **Refresh** to rerun the query to repopulate the data in the window.

Date Columns

- When you go into one of the update modes, any date / time / timestamp columns will be changed from their **display-format** to their **update-format**. If your timestamp column has a zero time-part (which is normally suppressed during the display of the column), you will notice that this has been added to the column. This allows you to update this time part of the column.
- For Oracle, you will get an option **Add To_Date**. This specifies whether you want the TO_DATE function to any date values being updated. If you click on **Show Quotes** you will see the column value with the To_Date function included.

Show Quotes

This option can be used to give you more control of the values you are updating or inserting. Normally, when you specify a value for a character column, AQT encloses this in quotes before updating the column with that value. When you have Show Quotes set on you will see the values as they will be used in the SQL statement.

There are number of circumstances where you would want to use this:

- if you are wanting to set your column to a system variable. Examples: SYSDATE, USER.
- if you are setting the column to another column, or a function involving another column. Examples: Current_Salary/100, Substr(Last_Name,1,1), Units_Ordered – Units_Delivered.
- if you want to use a parameter marker.
- you want to specify a value of NULL.

If you don't have Show Quotes set on then, for instance, a value of USER will get loaded into the column as the string 'USER' rather than the userid, **?** will get loaded as '**?**' instead of being interpreted as a parameter marker etc.

Running a Change

- to run your change, click on the Update / Delete / Insert button.
- if you change more than one row, and you have safe update mode selected, you will be prompted as to whether you want the change to proceed or not. This allows you to back out the change should it affect more rows than you expected.
- alternatively, before running your change you can click on Transaction to enter transaction mode. This will enable you backout your change should you decide that it didn't do what you want.

Preview SQL

Before running you change you can click on **Preview SQL** to see the SQL statement that will be run. While in the Preview window:

- you can modify the SQL.
- click on **Run** to run the SQL.
- click on **Save Query** to save the SQL. **Retrieve Query** retrieves a saved query.
    - if you change this SQL but wish to return to the original version of it, click on **Refresh SQL**.

0 Check Number

1 Before running an Update or Delete you can click on **Check Number**. This will run a query against the database to tell you the number of rows which will be affected by your Update/Delete. This is a useful safeguard before running a complex or critical update.

2 If you have a large table, this option could take a few seconds.

Running a Query
From the Table Info window, clicking on **Run SQL** will take you into the SQL window.

New Features in V4.2
Version V4.2 offers the following new features:

- you can **right-click** the SQL text-box to get a menu of common options relating to your query.
- you can change the **font** used in the text box (Edit->Change Font)
- you can change the **case** of the selected text (Edit->Change Case)
- you can switch to another table without leaving the Window. Click on Table->**Switch to Another Table** (or Ctrl-T, or the Table Icon) and you will be taken to a window where you can specify another table. Select one then click on **OK** to return. Alternatively you can click on **Copy Table Name** to just copy the table name. You can then paste this table name into your SQL.
- AQT remembers the last 10 **tables** you dealt with. You see these in the **Table** menu item. Note: these are
- AQT remembers the last 10 **queries** you dealt with. You see these in the **Query** menu item.
- **Check Syntax**. There is an option to check the syntax of your query. You do this with Edit->**Check Syntax** (or **F6**). This does a "prepare" of your SQL against your database. This option does not work with some databases (depending on whether "deferred prepare" can be disabled).

New Features in V4.1.3
There is a new option **Table->Build Column List for Query Table** (also **F2**). This is useful if the table in your query no longer matches the column list (for instance you have retrieved a saved query). When you select this option, AQT will parse your SQL for the table name and regenerate the column list for this table.

- if your query has more than one table, you will be shown a popup menu of the tables in the query. Select the table you wish to have the column-list for.
- you can highlight a table-name in your SQL before clicking on F2. This will build the column-list for that table.

New Features in V4.1.2
With V4.1.2 there are a number of extra features in this window, which will be detailed here.

- There is a new button **Get Constants**, which gives you a number of sample constants for the column-type of your selected column. This constant would normally be used in a WHERE clause. This button is mainly useful for Date/Time/Timestamp data types, as it gives the in-built date/time constants, and shows you the format of date/time literals.
- Once you have selected a column in the column-list, you can right-click to get a list of **sample functions** for that column. Selecting one of these will add the column plus function to the SQL.
- You can now select multiple columns from the columns list (either by dragging the mouse down, or by using ctrl and shift). Once you have selected multiple columns, right-click will allow you add all the selected columns to the SQL (comma-separated). This is a very fast way if adding a group of columns to the SQL.
- You can now do multi-columns analysis on your table; select the multiple columns, then use the Analyse Column buttons.

Using the Run SQL Window
There are a large number of things you can do with this window…..

- to run an SQL query, type it into the panel and click on **Run** (or hit Enter(*)).
- you can run action SQL statements (Update, Create, Grant etc) as well as queries (Select).
- in the lower panels you have a number of common SQL Phrases and characters, plus the names of the columns in your tables. Clicking on these will add them to your SQL. Except – to add a column name you need to double-click on it (a single-click just "selects" it). (**)
- once you have "selected" a column (by clicking on it), you can click on **Get Column Values** to get all the distinct values the column takes. These will populate the right-hand panel. Clicking on one of these will add it to your SQL statement (for instance as part of a Where clause).
- If you want to put a line-break in your query, don't hit enter as this will run your query. Instead click on <newline> or use shift-enter (this is a Windows standard). Alternatively, switch off the Run=Enter behaviour of this window (see * below).
- You can edit your query using a normal editor. Click on **Edit->Edit SQL using Editor**. By default this will edit the query using Notepad, however you change this with Options->General Options.
- If a query is taking a long time you can cancel it by clicking on the **Abort** button. For this to work you must have switched on the **Async Query** option. Also your database must allow queries to be cancelled. See the discussion in Statement Options.
- Clicking on **Output To** allows you to send the result of the query to a **file** or to the **SQL Window**. The latter is useful if you are running an SQL statement which is generating SQL statements.
- you can click on the Query Wizard icon to get a more comprehensive facility for building a query.
- there are 20 levels of undo / redo for correcting typing and clicking errors.
- you can click on the **Reset** icon to quickly revert back to the "Select * from table" SQL.
- AQT maintains a history of queries run. Click on **Window->History**, the history icon, or Ctrl-h. This will show you the SQL History window. This history is saved to disk so will be retained between AQT session.
- You can redisplay the results of the last query you ran by clicking on **Window->Results** or Ctrl-r. For this to work you must have closed the Table Display window with the Close menu item (which actually just hides the window).

(*) You can switch off the Enter=Run behaviour of this Window. See Statement Options.
(**) You can change the entries in the listboxes; to add a new entry, right-click on the listbox, to delete an entry click on the entry then hit delete.

Once you have run your query you will be shown the Table Display window.

See also:
- Analysing your table
- Saving / Retrieving Queries
- Adding Parameters to your query
- running an Action Statement
- running multiple Action Statements
- Transaction mode
- running stored procedures
- applying a search condition to your table

Analysing your Table

AQT offers a number of options to quickly analyse your table. These are available under the Analyse menu option, or a number of icons. These all work by generating an SQL statement which does the appropriate query.

- **Number of rows.** This shows you the number of rows in your table.
- **Number of distinct values of column.** This, and the remaining options, is only available once a column (or group of columns) have been selected. This option tells you how many different values the columns take in the table. For instance, if the columns are the key columns, the result will equal the number of rows in the table. If however the columns are the same for every row, the result will be 1.
- **Analyse column.** This is a more useful version of the previous query. It will show you all the distinct values the column (or group of columns) takes, and the number of times each value occurs. This provides a very simple way of quickly seeing the type of data held in a column / group of columns.
- **Duplicate values of column.** This shows you which column values occur more than once in the table. This is often useful if you have a column (or set of columns) which you think should be (mostly) unique – and you want to see which values are not unique.

Notes:

- Don't try these options on very large tables or it can take a long time! Common sense should prevail.
- " Number of distinct values of column" doesn't work for some databases (Access is a good example), as it doesn't support the SQL statement we generate.

Applying a Search Condition to a Table

AQT allows you to apply a **search condition** to a table. This allows you to display particular rows in the table without having to code SQL.

You do this as follows:

- display a table
- from the table display window, click on **View->Find Rows** or the **Find** icon. This will take you into the Find Rows window

Within the Find Rows window you will notice the following:

- the current search condition (if there is one)
- a set of controls where you can specify a new search condition

You can either:

- amend the existing search condition, or type one into the "current search condition" box
- build up a new search condition in the "new search condition" frame by selecting the column name and value you wish to compare it against.

If you specify a new search condition you can either:

- use that as your search condition by clicking on the **Use this search condition** button. In this case the current search condition is replaced with the new one.
- use this search condition in addition to the existing one by clicking on the **Add to existing** button.
- when you use the "Add to existing" option, you must specify whether this condition makes the search **more restrictive** (eg. will show you less rows) or **less restrictive** (eg. show you more rows).

Other Buttons:

- **Clear** clears the current search condition (so that all the table will be displayed).
- **Reset** returns the search condition to what it was when you entered this window.
- **Get Values** is used to get the existing values for the selected column. This option may not work reliably if your query is a multi-table join.
- **OK** returns you to the Table Display window with your new search condition.
- **Cancel** returns you to the Table Display window with your old search condition.

Other notes:

- when you enter this window, you will have a search condition already set up in the "new search condition" frame. This condition relates to whatever cell in the table-display you clicked on before clicking on the Find icon.
- if you want to specify more complex searches, you need to use the **Run SQL** window, though this requires some knowledge of SQL.
- your search condition is now part of the SQL statement which is "behind" the table display. If you return or go to the SQL window you will see this search condition in the SQL statement.

Saving or Retrieving a Query

To Save or Retrieve a query, click on **File->Save Query** or **File->Open Query** (or Save/Open icons) from the Run SQL window.

This will take you to AQT's version of the Save/Open dialog. This is much like the standard Microsoft Save/Open dialog, with a few extra features.

- as you click on files, the contents of that file will be displayed on the right panel. For retrieving, this makes it easy to find the query you are looking for. For saving, this makes it easy to compare the query you are saving with an existing one.
- you can edit an existing file by clicking on the **Edit File** icon. By default this will edit the file using Notepad, however you change this with Options->General Options.
- You can delete or rename a file with the **Delete File** or **Rename File** icons.
- You can refresh to contents of the window by clicking on the **Refresh** icon.

Large Files

You cannot retrieve a file larger than 32K. If your query is larger than this you will need to use Run from File.

Query / Home Directory

Your default / home directory for saved queries is specified by in Option->File Locations. This will be the directory you will be taken into the first time you save / retrieve a query in your session. If you change to another directory you can quickly return to this default directory by clicking on the **Go to Home Directory** icon.

Clicking on the **Set this as Home Directory** will set the current directory to being your Home Directory.

Running Stored Procedures

Stored Procedures can be run in AQT in two ways.

In the Run SQL window, type the syntax for running a stored procedure. Example:

exec sp_who

exec sp_addlogin

This will run your procedure, but has a number of limitations:

- only SQL Server and Sybase support this syntax for running stored procedures
- it can only deal with input parameters (not output and input/output parameters)
- it only returns a single result-set

For a more comprehensive facility for running stored procedures, use the Run Stored Procedure feature. To use this:

- in the Table Information window, click on Procedures in the Database Explorer
- having selected a procedure, click on Tools->Run Procedure, or the Run Icon

This will bring up the Run Stored Procedure window.

Run from File

This option is used for running a large SQL script; the Retrieve option has a limit of 32K due to the size limit of the SQL window. When you use Run from File, the SQL statements are run directly from your file, without first being written to the SQL window.

When you select Run from File:

- text such as the following is placed in the SQL window: **<file>bigfile.sql <startline>1**
- you can, if you wish, change this to another file, or use a different value for startline. In this way, restarting from a particular point in a file is simple.
- run the file by clicking on Run or hitting enter (as per normal).
- the SQL Log window will be displayed showing the progress of the SQL statements. When you run from a file you will get an extra column in the grid – Line Number. If a statement fails, Line Number makes it easy for you to find it in your file. Also, it enables you to know what line to restart the script on.

Adding Parameters to your Query

Your query can include **substitutional parameters**. When you code these you will be prompted for these values every time you run the query. This can be very useful when setting up "generic" queries for saving and using later.

To include a parameter, simply code a **?** where you would code a data value in your query.
Example: **Select * from SCOTT.DEPT where DEPTNO = ?**

- never put the ? inside quotes, even if it is a character value.
- you can only use a parameter where you would use a data value. You cannot use it to substitute for a table or column name. For instance you cannot say **Select * from ? where DEPTNO = 10**

Running a Query with Parameters

When you run the query you will be shown a window where you can enter the **parameter description** and **parameter value** for each parameter.

- The **parameter description** will be saved with the query and is the text used to prompt for the parameter. In this example you might enter **Department Number** for this.
- The parameter value is the value for the parameter. For character strings you should not put quotes around this.
- When specifying a parameter value you cannot use the name of another column or a special value / register (such as NULL, CURRENT DATE etc). These will be interpreted as a character string and will not have the desired result.
- This screen also shows you the **parameter type** (character, decimal etc). This is for your information only.

Parameter Types not Available

Some databases, in particular Oracle, do not provide information about the **parameter types**. In this case the Parameter Type will be blank and you will get the message "Your database does not report on the Type of your parameters."
Some versions of the Oracle ODBC driver **do** provide parameter information – but it is varchar(999) irrespective of what the parameter really is. Oracle has decided that providing incorrect information is an improvement on providing no information.

Multiple Action Query

If you are running a stream of multiple statements, each which have parameter markers, AQT treats these in a slightly different way:

- AQT will prompt for the parameters for the first statement
- for subsequent statements it will not prompt for the parameters but will use the ones which were supplied for the first statement.

This allows you to run a set of action statements which deal with a particular set of data without being continually prompted for the parameters.

For some more technical information on parameters see technical discussion on parameters.

Comparing the Definition of a Table Across Two Systems

In large sites it is common to have the same tables defined in either in multiple systems (development, test, production etc) OR in multiple schemas within the same system. Often one needs to know whether these tables have the same definition, or, if not, what the differences are.

AQT provides a simple tool for comparing the definition of a table across two systems or schemas.

Within the Table Information window click on **Tools->Compare -> Set Options** or the compare icon.

- you will shown a dialog with the Table Compare options. You use this to specify whether you are comparing tables in **Another Database** or **Another Schema** (or both).
- if another database, select **Another Database** and click **Sign On** to sign on to the database you want to compare your table definitions to. After you have done this you will have the message "Second system is ……." on the Table Info status bar.
- if another schema, select **Another Schema** and enter the Schema name.
- click on OK, you will be shown the Compare Table Definition window.
- you can return to the Table Compare options window at any time by selecting Tools->Compare -> Set Options.

Compare Table Definition Window

- This window shows definition of the table on both systems side by side.
- The right-most column is Compare. If there is any mismatch in definitions of a column this will have something in it (eg. *Type* or *Len*). This indicates the first difference it has found between the two column definitions. If the column exists in one table but not the other you will get *Extra* here.
- the status-bar at the bottom of the window will report on whether the tables compare OK or not.
- if there is a mismatch your PC will also beep.
- by clicking on **Prev** or **Next** you can move to the prev/next table in your table list. By using these you can very rapidly compare all your tables.

Other comments:

- this system only compare only compares name, type, length, scale and nulls. Other column attributes (such as default, description) are not compared.
- it is possible to compare across database types (eg. a table defined in Oracle to the table defined in DB2). However this is generally not all that successful due to the numerous differences in data types between different databases. However seeing the definitions side by side makes it easy to do an eyeball comparison of the definitions.
- the compare works with views as well as tables. This just compares the definitions of the view columns (not the actual view definition).
- when comparing tables, the columns are sorted by column number. If you add a column into the middle of a table, Compare will flag a mismatch for this and every subsequent column. It is not smart enough to work out that the only difference is one added column.

Security Helper
The Security Helper is activated by clicking on **Tools->Security Helper** from within the Table Information window.

This window is designed to assist with managing the security of your users and tables. All databases have slight differences in the way security is implemented (eg. what rights can be granted, miscellaneous variations in the Grant/Revoke commands, the way groups/roles are implemented). AQT has flexibility to handle most of these variations.

The security helper is oriented towards granting a user access to a number of tables. The design of this window reflects this.

Specifying the User
When you activate this window you will be asked to specify the user you are dealing with.

- enter a user (or a list of users separated by commas).
- alternatively click on **Select From List**. This will show you a list of valid users. Select the user(s).
- for Oracle users, **Select from List** will first ask you whether you want to see Users or Roles. Select the relevant option (note that unless you have a high level of authority you may not be able to see Roles).
- if are granting access to a new user, for Oracle, Sybase and MS SQL Server you must first create that user (this isn't needed for DB2). You must do this before you enter this window.
- you can use the menu option **Show** to see the existing authorities for the user. This will show you the access list in two formats – without grantor and with grantor (the latter is useful if you are trying to revoke access rights).
- **Show->Existing Authorities** (with grantor) will show you the table access grid.

Specifying the Tables
You must next specify which tables the user(s) are going to have access to. This is done with the options in the middle of the screen.

- the box will be populated with the tables in your schema. Click on one of these to add this table to the **Accesses to be Granted** grid (this is described later).
- use the up/down arrows to select multiple items, or use the **Select All** button.
- the **Mask** box specifies what access rights on these tables are being granted. Select these before you select the tables (though you can correct this).
- the **With Grant Option** specifies whether you want this clause added to the Grant statement (this allows the user to grant the access rights to others).
- another way of granting access rights is to copy the access list from another user. This is a good way of setting up a new user on a system to be like an existing user. Click on **Copy access list from another user** (Oracle users will then be asked if it is a User or a Role they are copying). Select the user – their access rights will then populate the **Accesses to be Granted** grid.
- **Copy access list from another user** is also useful if you are wanting to revoke all the existing authorities from a user. Select the user you wish to revoke, then click on Revoke.

Accesses to be Granted grid

This grid shows you what accesses are to be granted on the user. The previous section described how to populate this grid. You can amend these entries as follows:

- clicking on one of the cells containing an access right will cycle the access right from N->Y-G. You can use this to change the access right you are granting.
- clicking on the Schema or Name will delete that row.
- you can do multiple rows in one go by selecting multiple cells.
- **Clear Grid** will remove all rows.

Granting Access

Once you are happy that the Accesses to be Granted grid contains the accesses you wish to grant to the user(s):

- click on **Grant** to grant the accesses (or **Revoke** (note) if you are wishing to revoke instead of Grant).
- the SQL Log window will be displayed showing the progress and status of the Grant / Revoke commands.
- you have the ability to customise the format of the Grant / Revoke commands. This is useful if you wish to include a particular parameter in these commands. At the bottom left of the screen is a box giving the format of the Grant / Revoke commands; amend these to include the parameter. Example: **revoke ? on ? from ? by all**.

Note that throughout this system we use the term **user** to refer to an id to which you grant table access rights. This will be a User, Group or Role.

The list of access rights on this box are (most) of the valid ones for your database. This list has been customised for your database.

When you are revoking accesses, be aware that most databases only allow you to revoke the accesses rights that you have granted. If someone else has granted the rights you may not be able to revoke them. DB2/390 can get around this problem; it has a parameter BY ALL to revoke the rights irrespective of who granted them.

Inserting a BLOB/CLOB from a File
With AQT you can load a BLOB or CLOB value from a file.

BLOBs (Binary Large Objects) are used to hold data such as images and sound files. **CLOBs** (Character Large Objects) are used to hold large amounts of text.

This feature also works for **IMAGE, LONG VARBINARY, TEXT, NTEXT** (Sybase, SQL Server), **VARCHAR**, **LONG VARCHAR** (DB2), **VARCHAR2**, **LONG** (Oracle) and **MEMO** (Access).

These column types are generically call LOBs.

There are two reasons why you may wish to load a LOB column from a file:
- LOBs are often very large, so are difficult to load via normal SQL
- BLOBs often contain hex data, also difficult to load using normal SQL

To load a LOB from a file, you specify a parameter marker (?) in the insert/update statement where the LOB is to go. Example:
insert into blob_table (keycol, blobcol) values(1,?)

Alternatively, when using **Insert Row** or **Update Row** window, specify a ? in the Lob column value.

In both these cases, when you run the SQL you will be shown the **Enter Query Parameters** window. There will be a button **Load from File**. Click on this to specify the file you wish to load into the LOB column (the text **<aqtfile>filename** will appear in the parameter value).

The **Load from File** button will appear automatically for parameters of types BLOB, CLOB, IMAGE and CHARACTER/TEXT greater than 1000 bytes. You will not see this button in Oracle (as this does not give the parameter type). Even though this button is not present you can still load the column from a file. You do this by typing **<aqtfile>filename** into the parameter value.

Note for DB2/UDB. When this feature is used with VARCHAR and LONG VARCHAR columns it sometimes gets a "truncation error" message. The reason for this are not yet understand.

Writing BLOB/CLOB Values to Files

BLOB and CLOB columns are difficult to display as:

- they are often very large, potentially several megabytes
- BLOBs often contain hex data, which is difficult to display meaningfully in the normal results grid

BLOBs and CLOBs are known generically as LOBs.

In addition, BLOBs and CLOBs often need to be sent to an external application to be displayed. For instance, an image file held in a BLOB column will need to be displayed using an image-viewer such as MS-Paint. A Word document will need to be sent to MS-Word.

To deal with these requirements, AQT has an option to export LOB column to files. This option can be selected by **Options->Display Options->How to Display Lobs**. Select **Send to Files**.

You can also select what column-types this is done for: BLOBs, CLOBs and LONG VARCHAR. To explain what these types mean:

- **BLOBs** means BLOB (DB2/Oracle), IMAGE and LONG VARBINARY (Sybase, SQL Server)
- **CLOBs** means CLOB (DB2)
- **LONG VARCHAR** means LONG VARCHAR (DB2), CLOB and LONG (Oracle), TEXT and NTEXT (Sybase, SQL Server) and MEMO (Access)

The **Send to Files** options are not retained between AQT sessions as it is expected that you would only have them switched on for short periods of time.

When you use this option:

- every LOB value will be written as a separate file
- when you display a table / result-set, the name of the LOB-file will appear in the display
- when you double-click a LOB-file-name it will show you the **Detail of a Data Value** window. The first 1000 bytes of the LOB-file will be shown. You can use **Show As** to display the LOB using a particular program.
- alternatively, from the results-grid, right-click a lob-file-name and use **View As**.

Other Notes:

- the BLOB files are stored in directory **lobs** within the AQT install directory.
- if you return a large result-set, or have very large Lobs, then you can fill up your disk very fast. Use this option with a degree of caution.
- the contents of the lobs directory is cleared out every time you switch on the **Send to Files** option.
- the system will run slower with this option switched on. Not only are you creating a lot of files; but you will also be pulling a lot more data from your database server.

Yet More Notes

- this feature has only recently been implemented. It will be enhanced in subsequent releases. If you use LOBs in your database we would appreciate some comments on how you would like to see them handled in AQT.

- the Row Detail window does not (yet) display LOB values. In the next release we will amend this to display the LOB files, plus edit and create LOBs (by linking to the appropriate external application).
- if you are an Oracle user, you need to use the Oracle ODBC driver rather than the Microsoft ODBC Driver for Oracle (this does not support BLOBs).

Running an Action Statement

Within the Run SQL window, you can run action SQL statements as well as queries.

- any valid SQL action statement can be run – Update, Delete, Insert, Create, Drop, Grant etc. Stored Procedures can be run.
- when an action statement is run you will get a message indicating the success (or otherwise) of the statement and the number of rows affected.
- you might consider running the statement under Transaction Mode so you can rollback the changes should you decide they haven't done what you have intended.
- you can run multiple statements (separated by a semicolon or other delimiter). See Running Multiple Action Statements for a discussion on this.
- Db2/UDB has the distinction between SQL statements (Select, Update, Grant etc) and Utility statements (Load, Export, Runstats). AQT can run both of these; to run Utility statements you need (in the Run SQL Window) to select "Run as Db2 Command".

Run Stored Procedure

The Run Stored Procedure window is invoked from the Table Display window. Select a procedure then click on **Tools->Run Stored Procedure**.

You will be shown the parameters for the Stored Procedure. These will be of types IN, OUT or INOUT.

- you can enter values in the IN and INOUT parameters
- if nothing is entered in a parameter, it is interpreted as NULL

Run the procedure by clicking on RUN.

- if the procedure returns a result-set you will be shown this.
- if the procedure returns multiple result-sets you will be these one followed by another. In V4.3 we will provide a facility for viewing these multiple result-sets at the same time.
- you will then be returned to the Run Procedure window. If the procedure has returned any OUT or INOUT parameters, you will be shown these values.

Click here if you are having trouble running Stored Procedures with Sybase.

Stored Procedure Problems with Sybase

AQT sometimes has trouble running Sybase stored procedures, depending on:

- the version of Sybase you are accessing
- the ODBC driver you are using
- whether the ODBC stored-procedures have been implemented correctly in the Sybase Server

The problems AQT has with stored procedure is related to the way AQT determines the **parameters** for your stored procedure. To get this information AQT uses the ODBC function **SQLProcedureColumns**. AQT is dependent on this function to work correctly, however for some Sybase implementations this function gives wildly incorrect information. Even on systems where this function works most of the time, it occasionally gives the wrong information.

Note that the Sybase ODBC driver implements the SQLProcedureColumns function by a stored procedure which get installed on your Sybase Server. This procedure will be installed by your system administrator as part of the setup of the ODBC driver software. If you are having trouble with Run Procedures, you could check with your system administrator that this procedure has been set up correctly, and is the correct version for your ODBC driver.

AQT offers two mechanisms for dealing with these problems:

- in the Run Procedure window you can change the parameter type (IN/OUT/INOUT) if it appears to be incorrect. This parameter type is one of the main things which can be incorrect. To change this; click on the type (IN) – it will then cycle through IN/OUT/INOUT.
- if your parameters are wildly inaccurate, you can specify that SQLProcedureColumns is not used; instead an **AQT query** is used to obtain the parameter info. To specify this, you need to edit your sybasee.cfg (or sybasee1103.cfg) file to have a line with '**aqtprocparms** (note the single quote at the start of this) in the file.

If you use **aqtprocparms**, AQT cannot determine the parameter type so it will be defaulted to IN. You can change this by clicking on the IN value (as described above).

Amending Individual Rows

Besides running action queries, Advanced Query Tool provides another mechanism for updating/deleting/inserting rows. This is done using the Row Display window.

To use this:

- display the contents of your table. See displaying tables if you wish to know how to do this.
- click on the row you want to update/delete then hit the "Update row" (or "Delete row") icon.
- You will be taken into Update Mode in the Row Display window. How to use this window is described in the Row Display help.
- clicking on **Preview SQL** shows you the SQL which has been generated for your update/delete/insert. You can click on **Save Query** to save this. Thus this window can be used as a mechanism for easily building update, delete and insert statements.

Inserting a Row Similar to another One

Often when you are entering a lot of data (test data etc) you would like to take an existing row and insert another row just like it, with only a couple of columns different (such as the key field and maybe one or two other columns). Inserting data in this way saves on a lot of typing.

AQT allows you to do this. Within the Row Display window one of the modes is **Insert Dup**. This generates an insert statement based on the values in the row you are currently displaying. You can then change the columns you want to be different then click on **Insert**.

Comments on Updating in AQT

Many other database products allow you to update rows simply by overtyping the data values in the table display.

With AQT we have deliberately avoided using this approach. This is because with the overtyping approach is that it is difficult for the user to know exactly how the update is happening; in particular a) what was being done b) how it was being done c) when it was being done.

Instead we give the user far better control of updates (though admittedly it makes AQT slightly more complicated to use).

Within the Update Row function, you can:

- use the Preview SQL button to see the exact details of how the update is being done
- the update is only done when you click on Update / Delete
- you can run the update under transactional control, so you have the option to backout the update if it doesn't do what you expect

In addition:

- you can control what rows get updated – you can update a single row or a set of rows equally easily

None of these can be said about the "overtyping" style of updating.

Options
One can set a number of options in the system.

These are divided into five categories:
- General Options
- Statement Options
- Display Options
- File Locations
- Config Files

If you change any options, clicking on **OK** will save them. Clicking on **Cancel** will exit this window without saving them.

The **Defaults** button will set all options (in all tabs) back to the default values (except the File Locations and Config Files options).

Most options are saved (in the Windows Registry) – any changes will be in effect the next time you run AQT. A few options are deemed to be related to your session so are not saved between sessions (eg. Debug, Varchar Columns).

Advanced Customisation

This section covers customising Advanced Query Tool by modifying the contents of the config files. The config files contain the SQL statements that control the system-table query feature of AQT.

You might wish to this is if you want to:

- change some of the system-table queries which have been delivered with AQT. This could be because you make extensive use of a particular feature of the database that you would like reported in the Table Information window. Alternatively you might like to disable a feature you do not wish to use.
- use AQT against a database which AQT hasn't yet been configured for. This would involve developing a config file for that database. If you wish to do this please contact us – we will be keen to help you with this.

While the config files can be altered, we do not recommend you do this, or at least you should do so only with caution. As a result we don't provide full tools or documentation for helping you with this. Most of what exists (detailed below) is for use by AQT development and not intended to be used by yourselves.

- file readme.cfg in the program directory gives comprehensive information about the entries in the config files.
- in general the config files are amended by using a text editor.
- there is an online tool for amending the config entries. This is reached by **Tools->Configure System Queries** from within the Table Information window. See System Query Config for more information on this.

More disclaimers; when AQT reads the config files it does some checking of the contents, but does not check everything. If there is an error in your config file, or some of the entries do not match up, you may only notice this by the system behaving strangely or giving obscure error messages. Using **Options->Debug** can provide you with more information about what is happening under the covers.

Plus:

- before you change a config file, make sure you save the existing version of the file.
- if you create a config file for a new database type you will need to make an entry in the Options->Config Files before it will be picked up.
- if you download a new version of AQT your config files will be overwritten. It is up to you to manage these and reapply your changes to the new files.
- we will be continually updating the config files and posting them on our web site (www.querytool.com). If there is a feature you really want, you could contact us so we can add it into the "official" version of the files.

System Query Configuration

This window shows the configuration of the system-table query component of AQT. In particular these entries contain:

- the text of all the SQL statements used to access the system tables
- the definition of the menu structures used to present this information

Before changing anything here you should read the comments in Advanced Customisation.

The file readme.cfg provides full information about the entries in the config file; you should read these before attempting to change anything through this screen.

The config file contains the following entries:

- **Database Explorer**. Determines what objects are displayed in the Database Explorer.
- **menu2**. Determines what options are displayed in the list-box at the top of the middle-grid (this list-box may not be apparent as it isn't displayed when there is only one entry). This controls what data displayed in the middle grid when an object in the Database Explorer is selected.
- **menu3**. Determines what options are displayed in the list-box at the top of the right-grid. These options are also displayed as a pop-up menu when you right-click an object in the middle grid. These options controls what data displayed in the right grid.
- **menu4**. Determines what pop-up menu is displayed when the user right-clicks on the right grid. Note: these entries can only be set up by manually altering the cfg file; there is no facility on the Customise Config Files for setting these up.
- **queries**. These are the actual SQL used in the system-table queries. Included in this are the names and widths of the columns in the resultant grid.

Below these entries are explained in more detail.

Database Explorer

This contains columns:

- **Caption**. This is the text of the entry in the menu list.
- **Query**. The number of the query that populates the child objects in the Database Explorer tree.
- **Menu Set**. Specifies which set of menu2 and menu3 menus are displayed when this option is selected.
- **Type**. A code to indicate what type of data is being displayed when this option is selected. This controls what icon is displayed with that object, and a number of other aspects of AQT operation. Takes values T (table), V (view), S (synonym/alias), Z (table, view or synonym depending on a type column), C (table, view or synonym but can't tell which), P (package), U (user), R (role), E (login), I (index), A (tablespace), B (procedure), G (trigger), L (plan), F (function), D (default), M (domain), O (other / unidentified).
- **Schema**. Whether the object being displayed has a schema/owner associated with it. Examples of such objects are users and tablespaces. For these, no child objects in the Database Explorer are populated. Takes values Y or N.

Menu2

This contains columns:

- **Title**. The Text of the menu item. This is the top-level menu caption.
- **Caption**. The text of the entry in the menu list.
- **Query**. The number of the query which populates the middle-hand grid. This query will be passed one parameter (which is the value selected in the left grid).

Note: if there is only a single entry in the menu2 list, the menu2 menu will not be displayed.

0 Menu3

1 This contains columns:

- **Title**. The Text of the menu item. This is the top-level menu caption.
- **Caption**. The text of the entry in the menu list.
- **Query**. The number of the query which populates the middle-hand grid. This query will be passed two parameters (the values selected in the left and right grids), unless menu1.schema='N' in which case it will be passed a single parameter.
- **Pop Menu**. The number of the pop-up menu which will be used when the user right-clicks on the right-menu.

Query Details

This contains columns:

- **Query**. The number of the query.
- **Type**. A code to indicate how this query is to be processed. For most queries this is M. See readme.cfg for a complete list of these and what they mean. This is no relation to the Type on the Database Explorer entries.
- **Title**. The title of the query results. This will have the object name appended on the right.
- **Parm Type**. A special flag used when object-ids (rather than object names) are used in the query. This is used extensively with Sybase Enterprise and MS SQL Server. See readme.cfg for more on this.

The text box contains the SQL text of the query. This is followed by a number of boxes where one can enter the Column Titles and Column Widths of all the data returned by the query. Some of the existing queries use "hidden columns" – these have a width of 1. Do not use zero.

Options - General Options
In normal circumstances you would not wish to change any of these options.

Table Info
This determines whether table information is obtained from the System Tables or from the ODBC API calls. The default is System Tables (if available for this database).

This parameter cannot be changed while you are currently signed onto a database.

For a further discussion on this see System-Table Query Feature.

Parameter Usage
This is quite a technical parameter. The short explanation is that you should only consider changing this if you are having severe problems with you database access. This option was introduced to circumvent problems we were having with some ODBC drivers.

If you are technical type you might want to read the technical explanation of this option.

Debug Mode
This is only used to collect diagnostic information for the AQT developers. You would switch this on if you are having a problem and are collecting diagnostic information.
The debugging information is written to a file (aqtdebug.txt in the install directory). There will be a performance impact while this option is switched on. **View** can be used to look at this debug file.

Statement Delimiter
This option is also used when running SQL statements. By default the semicolon is used to delimit multiple SQL statements. This can cause a problem if you SQL statement contains a semicolon (this can happen with procedure or trigger definitions). To prevent a semicolon from being interpreted as a statement delimiter, select either None (No delimiter) or Other (and specify a delimiter character)

For further information on this see Running Multiple Action Queries.

Quote Table and Column Names
In some circumstances, column and table names need to have double-quotes placed around them. This needs to be done if either the name contains a special character (such as a space or a comma), or if the name is a reserved word in the database.
By default AQT operates in **smart** mode, which means to only place quotes around the name if it contains a special character. However this isn't completely full-proof – it checks for some but not all special characters, and it doesn't check for reserved words.
Select option **always** to force AQT to always put quotes around table and column names.

Background Colour
Allows you to select the colour used in many of the AQT windows.

Run SQL

Specifies the behaviour of the Run SQL window when Enter hit. By default this will be interpreted as Run (the query). Unselecting **Run SQL when hit Enter** will switch off this behaviour; enter will be interpreted as <newline> in the SQL window.

Editor Program
Sets the name of the program to be used for Editing queries and SQL text. See Saving or Retrieving Queries. At this stage only Notepad can be used (this is because AQT needs to wait until the edit session is finished and, for reasons which are beyond us, this only works for Notepad).

Parameter Usage – Technical Description
Within AQT you can use parameter markers (eg. ?s) to indicate substitutional parameters. This feature is used extensively by the system queries that populate the Table Information window.

There are two ways AQT can process these parameters:

- **Use Parameters** pass the parameters to the ODBC driver (and database) for processing. This is the default and recommended behaviour. Parameter markers are a standard part of the ODBC interface; once the statement has been "prepared" the ODBC driver will return to AQT the number of parameters (and their type). AQT uses this information to prompt for the parameter values. These values are then set and the query executed.
- **Use Substitution** AQT itself to scan the SQL statement for any ?s and itself replace the ?s with the values which have been supplied by the user. No parameter markers are passed to the ODBC driver.

The second option should only be used if your ODBC driver does not support parameter markers or if you think there is some problem with them. You can test this by running a query such as **Select * from Table where Column=?** from the Run SQL window.

Parameter Type
When the SQL statement is sent to the database, the database replies with the number of parameters and their types. This type is displayed on the Parameter Prompt window. A number of databases return incorrect (or no) information on parameters; consequently they sometimes have trouble with queries with parameters. Numeric parameters seem to be a particular problem area. At worst, you will need to avoid these getting problems by switching to **Use Substitution**.

However Use Substitution itself has problems with parameter types. AQT has no knowledge of the type required (it just sees a ?), and it needs to know whether to place quotes around the supplied value. To partially get around this, if AQT notices that the parameter is numeric it will not put quotes around it, otherwise it will. You may then get a problem if you wish to pass numeric characters to a character field; in this case you will need to put quotes around the value.

System-Table Queries
The system-table queries use parameters extensively. In addition, MS SQL Server and Sybase Enterprise use numeric parameters, which seems to be a particularly buggy area. If your database has any problem with parameters you will probably notice that your system-table queries are failing with obscure errors. Try switching to Use Substitution, or switching off the system-table feature altogether with Table Info ->Use ODBC.
Alternatively you can sometimes code around parameter problems. For instance you could use **where id=convert(int,?)**. We use this throughout our Sybase Enterprise system queries.

Options - Statement Options

Isolation Level

This is a technical parameter which specifies how you interact with other transactions which are being done on your database server. This parameter is only relevant if you are querying a database which is being updated at the same time you are running queries. **Committed Read** (the default) means you only see rows which have been committed. **UnCommitted Read** means you see rows even if they have not been committed.

This parameter is mainly useful for DB2; with Committed Read a query will take "read locks" and can hang while running against a table which is being updated. In this type of environment, it is recommended that Uncommitted Read is used.

Uncommitted Read is not available on many databases and, if specified, will be ignored.

Async Queries

This option specifies whether your SQL statements are processed asynchronously. This issue is explained below.

The default behaviour (synchronous) means that AQT is suspended/hung while the query is being processed. You must wait until the result of the query returns from the database. This can be a problem if you (inadvertedly) do a query that takes a long time to process. The only way you can terminate the query is to kill AQT using the Windows Task Manager (or similar tool).

If you select Async Queries then AQT remains active while the query is running. The main advantage of this is that you will now have the ability to cancel the query, should you decide that it's taking too long. When Async Queries has been selected, the Abort button on the Run SQL window will be enabled and you can click on this cancel the query.

Other comments:

- async queries are not available for all databases. If you select this option you may get the message "Note: the database doesn't support async processing so this setting will be ignored".
- if you want to know whether or not your database supports async queries, see **Help->Database Details->Async Allowed?**
- a final but important comment. We have had a lot of problems with this option. This is possibly due to bugs with the ODBC drivers and/or the databases. Use this option with caution.

Safe Update Mode

This feature was introduced to prevent people inadvertedly updating or deleting more rows in a table than they planned on. At every site we've worked at, every now and then someone will accidentally update or delete all the rows in a critical production table. This feature will help prevent this.

With this feature enabled, AQT will warn you if your SQL statement affects (eg. updates/deletes) more rows than you "expect". The "expect" here is given by the parameter "Warn when more than x rows updated". If you do you will be given the option to proceed with the update or rollback the change.

By default the warn-value is set to 1, so any update affecting more than one row will get this

warning message. You can set this to a higher value if you are running a series of updates which you know are going to affect multiple rows, however we recommend that you do not switch this off.
From a technical point of view, this feature is achieved by doing a Start Transaction before any update, then a commit / rollback, End Transaction after. Some databases (such as Access) occasionally have trouble with Transactions and give errors such as "Option not valid at this time". If this happens you will have to switch Safe Update Mode off.
This options is not available in MySQL, as this database does not support transactions.

Date and Time Columns

Date and Time columns are a very complicated area of AQT as:

- people want the flexibility to view them in a particular format.
- most databases are very fussy about how dates are specified in insert/update statements.
- all databases have different date/time fields, and different rules about how they are handled.

Display Format versus Update Format

AQT uses two different formats for date/time columns – the **Display Format** and the **Update Format**. The Display Format is the way you wish to see the dates displayed. Whenever AQT is displaying a table, it will display the dates in this format. AQT will use the **Update Format** when you are specifying a date in an Update/Insert Statement or a in Where clause. This happens in two places:

- when you do a Get Values on a date/time column you will get it in the Update Format
- when you go into Update mode on a table AQT will changed the date/time columns from the Display format to the Update format.

You set the **Display Format** with Options->Display Options ->Format of Date/Time Columns. The **Update Format** is hard-coded within AQT, based on the database type (see the section Date Data Types below).

Date Data Types

The following discussion summarises the date / time / timestamp data-types of the various databases. In this:

DB2

Type	Stores	Required Format
Date	Date	yyyy-mm-dd
Time	Time	hh.nn.ss or hh:mm:ss
Timestamp	Date, Time to the microsecond	yyyy-mm-dd-hh.nn.ss.dddddd

Note that with the Timestamp column, the Date and Time parts are connected by a minus sign. This is unlike most other databases that use a blank.

Oracle

Type	Stores	Required Format
Datetime	Date, Time to the second	Date specified in the format of NLS_DATE_FORMAT. Time specified in ??? format.

AQT will set the NLS_DATE_FORMAT to the format you want to display the dates. This allows you to use the same format for displaying dates as for specifying in Where clauses (in other words, the Display Format and Update Format will be the same).

Sybase Enterprise and MS SQL Server

Type	Stores	Required Format
Datetime	Date, Time to millisecond	yyyy-mm-dd hh:nn:ss.ddd
SmallDateTime	Date, Time to the minute	yyyy-mm-dd hh:nn
Timestamp	?	

Informix

Type	Stores	Required Format
DATE	Date	mm/dd/yyyy
DATETIME A TO B (example: DATETIME Year TO Second)	Flexible, depending on column definition.	

MS Access

Type	Stores	Required Format
Datetime	Date, Time to second	yyyy-mm-dd hh:nn:ss

When specifying DateTime literals, Access requires these to be enclosed in #s (not single-quotes like every other database). AQT will do this automatically.

Date Functions

Many databases provide functions for specifying date/time values.

- in Oracle this is **TO_DATE('2001-12-31 15:32:55', 'yyyy-mm-dd hh24:mi:ss')**
- in Informix this is **DATE("12/31/200")** for Date values or **DateTime(2001-12-31 15:32:55) Year to Second** for Datetime values

In these databases it is normal to use these functions for specifying Date/time values. In some cases, date/times can only be specified using such functions. AQT can automatically include these functions when a date/time value to be given. In the Run SQL (and other) Windows, a Checkbox "Include TO_DATE" is shown when you are dealing with a Datetime column. If you select this, AQT will include the TO_DATE (or other) function with the Date/time value.

Dates in Parameters

In AQT you can use parameter markers in queries:

Select * from DEMO.EMPLOYEE WHERE HIRE_DATE = ?

When you run this you will be prompted for the value of Hire Date. You must specify a correct Date/Time value here. This value this must be in the **Update** format. Functions such as TO_DATE cannot be used here.

With Oracle there is a further complication with queries such as these – due to limitations with the Oracle ODBC driver, AQT does not know the "type" of the parameter. You will be shown that the parameter is Char(100) or Varchar2(999) rather than Datetime.

Suppression of Time

Oracle, Sybase, SQL Server and Access do not have a separate Date type. Dates are normally stored in Datetime columns, with a zero time-part. When AQT sees a zero time-part it will not display it; it will display the column as a date. In other words, rather than display the column as "2001-12-31 00:00:00", AQT will display it as "2001-12-31". If the time-part is non-zero AQT will display it. The one exception to this is when you are updating a date; in this situation AQT will include the time part in case you wish to update this.

Options - Display Options
These are options you might wish to change occasionally.

Max Rows Displayed
Specifies the number of rows to display when a query is run, or when a table is displayed. While you may just wish to set this to the maximum value, be aware that when you set this to a large value your queries will take a long time to run and more memory will be used.

Our recommendation is to set this to a low value (200) and just change this to a higher value on the (rare) occasions when you want to see a lot of rows.

Warning: if you set this to **No Limit**, and display a very large number of rows, your machine can run out of memory. Use this with caution.

If you are running the Evaluation version of AQT you will not be able to increase this beyond 50 rows.

Max Column Size
The issue here is similar to Max Rows Displayed. This option exists to minimise the amount of memory used to store the result of the query. This option determines the maximum size of a column held in the result-set.

Our recommendation is to set this to a low value (50 or 200) and just change this to a higher value when you wish to see the full text of your large columns.

No Limit actually has a limit 10,000 bytes. You can view larger columns than this by using the **How to Display Lobs->Send to Files** option.

Show Length of Varchar Columns?
This is a potentially useful option for database developers. When a value is inserted into a Varchar column the length of the column has to be set (in C and Java this is generally done with the null byte, in other languages a length field is set). When you display the table you can verify that the data has been loaded correctly, however the column length can not be seen.

When you select this option the length of the column is displayed along with the data. For instance FRED will be displayed as (4)FRED. This can be useful for verifying that the column lengths have been set correctly - if instead you see (20)FRED then you know that you are filling your database up with a lot of unnecessary blanks! This has happened on some projects we have been involved in.

How to Display Binary Columns
Binary Columns are columns which can contain non-ascii data (in DB2 these are defined as CHAR FOR BIT DATA, in Oracle they are defined as RAW, in Sybase and SQL Server they are defined as BINARY). This also includes BLOB columns.

AQT can display binary data in two formats:

- **ascii** – the ascii representation of the data is displayed. This may well give unreadable values.

- **hex** – the data is displayed in hex mode. For DB2 hex data is displayed like x'44', other databases display hex data as 0x44.

The default mode for AQT is **smart**. In this mode, AQT will display the binary columns in either ascii mode or hex mode depending on the contents of the column – eg. it is displayed in ascii mode if the column contains only displayable characters, otherwise it is displayed in hex mode.

When columns are displayed in hex mode only the first 50 characters are shown.

Show Nulls as
This specifies how null values are displayed.

How to Show Lobs
With the default option (**Display**), AQT will try to display the value of BLOB/CLOB columns in the same way other columns are displayed. This may not give useful results if the column contains hex data or is very large. In this case, the option **Send to Files** provides a better mechanism for displaying LOB columns. This is described in more detail here.
You can also select what column-types this is done for: BLOBs, CLOBs and LONG VARCHAR:
- **BLOBs** means BLOB (DB2/Oracle), IMAGE and LONG VARBINARY (Sybase, SQL Server)
- **CLOBs** means CLOB (DB2)
- **LONG VARCHAR** means LONG VARCHAR (DB2), CLOB and LONG (Oracle), TEXT and NTEXT (Sybase, SQL Server) and MEMO (Access)

Show ROWID
For Oracle and Informix users, this option will show the Rowid when the table contents are displayed.

Format of Date/Time Columns
These options specify how Date, Time, DateTime and Timestamp columns are displayed. Enter a format using a combination of the letters: y (year), m (month), d (day), h (hour), n (minute) and s (second) plus any delimiting characters (such as -,: or /).

Examples:
- mm-dd-yyyy will display as 12-31-2001
- dd/mmm/yyyy will display as 31/Dec/2001
- hh.nn.ss will display as 15.32.58

Other Comments:
- Timestamps and Datetime columns display as: Date Format + " " + Time Format.
- See Date and Time columns for more on Dates /Time columns.
- The **date** format MUST include the following specifications: yyyy, mm or mmm, dd.
- The **time** format MUST include the following specifications: hh, nn, ss.
- For Oracle, the **NLS_DATE_FORMAT** will be changed to what you specify for the Date Format.

Options - File Locations

These options specify the directory where the relevant files reside. In both cases clicking on **Change** allows you to change this directory. Clicking on **Default** will reset this to a value of blank, which means the directory where the AQT executable resides.

Location of Config Files
This specifies where the config files are located. See Database Types and Config Files. **for a description of the config files.**

Location of Saved Queries
This specifies the directory where (by default) queries will be saved / retrieved. This is used in the Save Query window.

Options - Config Files

The entries here are used to cross reference the database type to the config file to be used for the database. These entries are key to the correct functioning of the System-table Query Feature of the product. See also Database Types and Config Files for a discussion on this feature and how it relates to the config files.

AQT operates on the idea of auto-detecting the database type. This is not a trivial matter. It relies on what the database calls itself. This is reported in the "The database you are currently signed onto identifies itself as" field. AQT then tries to match this with one of the entries in the list of Database Identifiers.

AQT will populate the list of Database Identifiers when runs for the first time. If there are any problems with these entries you will notice that the system-table query feature is not functioning (eg. in the Table Information window the Database Explorer only shows Tables/Views and System Tables).

Circumstances where you might need to change these entries are:

- your database identifies itself with a name different to what we have configured.
- you have been supplied a config file for a new database type (either by us or by someone at your site) and you need to add in a new entry in order for this config file to be picked up.

The way these entries work as follows:

- given the name the database calls itself, AQT will go through the "database identifier" entries for the first one which matches.
- when matching an entry, AQT tries to match on the first n characters of the names, where n is the length of the database identifier entry.

An example: DB2/390 identifies itself as DB2, DB2 for AIX as DB2/6000, DB2 for NT as DB2/NT. These databases are handled by the following entries:

- DB2/ catches DB2/6000 and DB2/NT
- DB2 catches DB2 (for 390)

This would not work if these entries were in the opposite order (as the DB2 entry would catch all three databases).

There is no capability to specify different config file for different versions of a database.

Printing Data

Most windows in AQT have the option for **Print Screen**. This sends an image of the screen to the printer and as such is a basic print facility.

0 In other places there will **Print Data** or **Print Grid** options. This displays the **Print** window that provides a more comprehensive printing facility for printing a grid of data.

1 The Print window has the following features:

- Gives you good control about what data will appear on your page.
- Allows you to print multiple pages horizontally; useful if your table has more columns than will fit onto one page.
- Has a Print Preview Feature.

Controlling the Data Which Will appear on your Page

One of the main issues which face you when printing a grid of data (such as the result of a query) is that the data is generally wider than what will fit onto a page. AQT has two features that help here:

- AQT allows you to print multiple pages horizontally, so you can print all your columns.
- You have a lot of control about what columns appear on which horizontal-pages.

At the bottom of the window is a grid that shows you the columns in the grid you are printing.

- **Page (L-R)** shows you which horizontal-page the column will be printed on.
- As you change the page layout details (**margins** or **portrait / landscape)** you will see the page width change; the pages the columns go on also changes.
- **Start** and **Width** helps you understand how the columns are fitting onto the page. This gives the start position and width of the columns (the width of the page is given higher up in the Page Layout section). You can adjust these by changing the Width of a column.
- You can exclude a column from the display by clicking on the **Page (L-R)** value. When you do this **<excluded>** will be shown in that column. Click a second time to re-include it for printing.

Also on the print window are:

- **printer**. Select the printer here.
- **font**. Change the font and font size.
- **layout**. This specifies the page layout and shows you the number of pages which will be printed.
- **pages down**. Use this to select the number of vertical pages you are going to print. By default all pages will be printed. To change this, deselect All Pages and change the page value.
- **pages across**. Use this to select the number of vertical pages you are going to print. This defaults to 1.
- **footer**. This is the title of the report which is to appear at the bottom of the pages. When you are printing result of a query this will be the first part of the query SQL. You can change this to a more appropriate title if you want.
- **margins**. Specifies the page margins.
- **stripe** and **stripe freq.** Allows you to put "stripes" across your output. This can often make it easier to line up rows on a printout. To see what this is all about, select this then do Print Preview.

Print the report by clicking on **Print**. You can cancel the print while it is running with **Stop Printing** (how well this works depends on your print driver etc etc. No guarantees).

Print Preview
Clicking on this will show you the report as it will be printed. You can click on **Page Down**, **Page Right** or **Page Right** to move around the report.
There is no Zoom-in or Zoom-out feature. This will be put in if enough people ask for it.

General Useability Concepts

The following are some general rules about the way the end-user interface works. In some cases these are (slightly) different from the way other windows applications work. However our focus has always been ease-of-use and we feel we have avoided many annoying aspects of other systems.

- **Message Boxes**. These are not a lot in AQT. This is because having to click on OK to close these can sometimes be a real nuisance. Instead informative messages are written to the Status Bar at the bottom of the window. If the message is long and doesn't fit into the status bar, click on it to see the full message.

- **AQT Message Box**. On the rare occasions when we do display a message-box you may see AQT's own version of a message box. This has the advantages that it can be re-sized (in case it's displaying a lot of data), printed, contents saved or copied to the clipboard (these are all short-comings of standard message boxes which have always annoyed the hell out of us!).

- **Menus**. We've tried to avoid burying important actions and options inside menu structures. Where possible, the things you commonly wish to do in a window will have command buttons or an icon At worst will be a top-level menu item. In this way the action is very visible (you won't have to go hunting to find it) and can be selected with a single click.

- **Window Sizes**. Most windows can be re-sized. AQT will remember these sizes, even across AQT sessions. In the Table Info window the right divider-bar can be moved; this position is remembered too.

Hints on Grid Usage

AQT uses grids in many of its windows. The following are some general rules about things you can do with the grids.

- **Copying**. If you wish to copy cells (to the Windows clipboard), select a range of cells then hit Ctrl-C. You will be shown a menu asking you about various options related to the copy: you can select:
  - delimiter. The character used to delimit the cells. Select tab if you wish to paste the data into Excel.
  - include header. Whether the column header is to be included with the cells. This can be useful if the data is being transferred to Excel or Word.
  - replace / append. This option specifies whether the data is to replace or add-to the existing data in the clipboard. This can be useful if you are wishing to copy non-contiguous ranges to cells.
- **Printing**. The data in a grid can be printed by selecting the menu option **Print->Print Data**. See Printing Data for more on this.
- **Displaying wide columns**. We have (deliberately) set a limit on how wide a column gets. This allows you to see the other columns in the grid, rather than a single column taking the entire width of the window (this often happens with other query tools). To see the entire value of a column you can either increase the width of the column or (better) double-click on the cell. Throughout most of AQT, double-clicking on a cell will show you the full value of the cell.

Sending the Query Results to a File

This can be done by selecting **Output To->File** from within the Run SQL window. This will display the File Output window.

There are a number of options (most of these are fairly obvious):

- **File Format**. Specifies the format of the output file. There are a number of options here - comma delimited (csv), tab delimited, delimited with another character. Fixed format, which would be used to generate a file suitable for loading into DB2/390 (though with some limitations).
- **Output as HTML.** With this option, the data will be saved as an HTML table. AQT does not provide any options about the format of this table.... AQT is not designed to be an HTML editor. Once AQT has generated the table, use your favourite HTML editor to specify other formatting options (colour, border widths) and to add buttons, text and other objects to the HTML page.
- **Output as INSERTs.** This option will output the data as a series of Insert statements. These can then be used for loading the data into another table. This provides a very simple mechanism for transferring data between tables, especially if the tables exist on different databases or database types. When you select this option you will be prompted for the name of the table you wish to load the data into. You will also be prompted for the maximum line-length of the output file.
- **Include Column Header?** Specifies whether or not the first row in your output file is to contain the column names. This might be useful, for instance, when importing the data into an Excel spreadsheet.
- **Delimit Strings?** Allows you to specify a character to enclose string values. " is often used here. When using Column Delimiters it is safer to use this option (to avoid the problems you would get if you string included the delimiter character (eg. the comma)).
- **Max Rows.** The maximum number of rows to write to the output file.
- **Append to File?** Whether the data is to replace the existing file (No), or be written after the existing data (Yes).
- **File**. Select the file you want the data written to. Type in a value or click on **Change**.
- **Remove from columns**. A couple of things can cause your file to give very strange results when loaded into another database or Excel. The first is if you have a column that contains the character you are using as the string delimiter (eg. "). The second is if your column contains a line-feed character. The Remove from Columns option allows you to remove these characters from you columns. It is recommended that you leave these options on.
- **Show Nulls as**. This is also controlled by an option on Options->Display Options. In most cases you would want to have nulls shown as blank in an output file; this allows you to easily override the Display Options for this setting.

The **query** box reminds you of what your query is, in case you have a bad memory (I find this very useful). This cannot be changed.

To run your query click on **Run**.

- if you are overwriting a file you will be prompted about whether you wish to do so (unless you have select Append).
- while your query is running you will be taken back to the Run SQL window. You will be given

regular progress report on how many rows have been written to your file. If you wish to terminate the query, click on Abort.

Once your query has finished you will be asked whether you wish to view the file or not. If you do:

- you will be shown the first part of the file (first 32K bytes).
- you can copy data from this window by selecting the data then hitting Ctrl-C. Alternatively clicking on **Copy All** copies all the data.
- the **Font** option deserves mention. If you have written the data in fixed format you need to view the data using a non-proportional font (eg. one where all characters have the same width). Selecting **Font->Non Proportional** makes viewing of fixed format data much easier.

numeric variables are not padded on left with zeros (as they should be). Also it would be nice if AQT generated the control statements for the DB2/390 load utility (with column names and positions etc). If anyone wants us to add these features in please let us know.

Query Builder
In the Run SQL window, clicking on **Query Builder** will take you to Query Builder window. This provides you with a way of building simple Select statements (click here for a list of limitations of the query builder).

If you want to build non-select statements (such as updates or inserts), use the options in the Row Display window.

The query builder window has the following elements:
- the column grid; this is where you build your query.
- SQL text; this shows the text of your query.
- column info; this displays information about the selected column

As you build your query the SQL text is updated. By seeing how this SQL text changes when you select various options you can rapidly gain an understanding of how the query builder works.

Column Grid
You build your query by using the column grid. This grid has a row for every column in your table, plus a "spare" row. The grid has the following columns:
- **Column**. This is a column in your table. When you select a column (by clicking on it), the column info box shows you information about that column.
- **Show?** This specifies whether the column is to be displayed in your query. Clicking on this cell toggles between Y (show column) and blank (do not show column). The order of the columns in your query is governed by the order they appear in this grid. You can move columns up or down in the grid by selecting them then clicking on **Up** or **Down**.
- **Order By**. This specifies the order of the rows in the query – eg. builds an Order By statement in your SQL. Clicking on the cell gives the column a numeric value (1,2 etc) which is the position of the column in the Order By statement.
- **Group By**. This is used to build a Group By statement. This is described here.
- **Col Func**. This is used to specify a function on your column. When you click on this cell you will be shown a drop-down list of example column functions. Click on one of these or type your own. When you specify a column function, use a : (colon) where the column name is to go; AQT will do this substitution.
- **Op**. Specifies a search operator (for the Where clause). Click on this cell then select from the list or type in your own.
- **Search Value**. Specifies a search condition on the column. Click on this cell - you will have the option to get the existing values for that column by clicking on **Get Values**. Select a value from this list or type a value. You can specify multiple search conditions – AQT will put an AND between them (eg. will only select rows which meet both conditions). The query builder cannot build more sophisticated Where clauses.

Other Notes:
- spare row. The spare row in the column grid is used to specify functions which aren't associated with any column. An example of this is Count(*).
- distinct values. Click on the **Distinct** option if you want the clause Distinct added to your SQL. This ensures that your query only returns unique values.

- for Sybase and MS SQL Server you can select option **Select Into**. This adds the phrase INTO #temptab to your SQL, allowing you to create a temporary table containing the results of your query.
- removing the where clause. To do this just blank out the Op cell. You don't have to blank out the Search Value cell.
- menu option **Selection** allows you to quickly select or de-select all columns.
- menu option **Reset** returns your query to it's original state (as when you entered this window).

Interpreting your SQL

One of the more complication parts to the query builder is as follows. When you select Query Builder from the Run SQL window, AQT will take your existing SQL statement and try to interpret it. The idea here is to split the SQL text into column list, column functions, order by list etc; the column grid is then populated with these values in such a way that it represents the SQL statement which you had in the Run SQL window.

This is a very complicated thing to do, and there are distinct limitations on how well this works. If you have a complex SQL statement, AQT will not succeed in this; you may get some warning/error messages and the values in the column grid will not represent your SQL statement.

Group By

The Group By clause is used to summarise your table. Before using this function you need to understand how the Group By function works.

Take an example: if you wished to summarise your revenue by Customer and Month you would code an SQL statement such as:

Select Customer, Month, Sum(Revenue) from Rev_Table Group by Customer,Month

You specify in the Group By clause the columns you are summarising your table by.

The important rule about Group By clauses is as follows. The only columns you can have in your column list are:

- the Group By columns
- summary functions on other columns

Examples of summary functions are: Sum, Min, Max, Avg, Count(*).

Within the query builder you specify the Group By columns by clicking on the Group By column. The rule we have stated above means that: all columns you are displaying (eg. have Show? of Y) must be either one of the Group By columns OR have a (summary) column function specified for it.

If you do not meet these conditions then the query builder will give you warning messages (however it will still build the query as you have specified it).

The query builder is only designed to build simple queries. It cannot build:

- multi-table queries
- ORDER BY clauses which have descending order
- complex WHERE clauses
- sub-select statements

DDL Generator

DDL for your tables and views can be generated by clicking on **DDL** from the Table Information window.

This is a basic DDL generator. It is not designed to rebuild a complete production system; instead it is more designed for a database developer who needs to move table definitions between databases, or between schemas within a database.

It will not generate:

- keys (primary and foreign), indexes, column constraints, triggers on the table
- the "in tablespace" clause for the table
- " obscure" column definitions

To use the DDL Generator window:

- select which tables / views you want to generate DDL for by clicking on the Select column in the grid. **Select all** selects all objects, **De-Select all** de-selects them all.
- if your list of objects contains both tables and views, use the **Generate DDL for** box to select the type of object you wish to generate DDL for.
- in the **Options** box, select whether you want to generate column comments and/or default values. Note that for DB2/390 the system cannot generate default values successfully due to the complex way in which these are held in the catalog.
- the **Generate views as tables** option is interesting. As a developer you may be wanting to create a copy of a view on your own database in order to test a program you are developing. Normally AQT will generate the view DDL as per the original Create View statement – this might be a complex query involving multiple tables and views. This view can only be created by first creating all these base tables/views. This can turn creating the view into a large job. Instead, when you select **Generate views as tables**, AQT will generate DDL for a table which looks just like the view. Not only can this be easily created, but by having a table (instead of a complex view), test data can easily be loaded.
- **native or user-defined data types**. Specifies whether the columns are defined using the native data types (char, integer etc), or user-defined types (SQL Server and Sybase Enterprise only).
- **output to** specifies where the generated DDL is to go to (a **window** or a **file**). The window option is useful for quickly seeing what is being generated (note that the window has a limit of 32K so cannot be used for a large number of tables).
- **include schema name** specifies whether the schema/creator/owner name is to be included in the Create Table/View. You can choose a particular schema name to be used, or none at all (in which case the objects will be created under the id of the person running the statements). Note that in the case of Create View statements, the schema names of any objects referenced within the view-query are not affected by this parameter.

To generate the DDL, click on the **Generate** button:

- if **output to window** was selected you will see the DDL. You can copy this (to the clipboard) for pasting into a file or the Run SQL window.
- if **output to file** was selected the DDL will go to the file. You will have the option of viewing the (first part) of this file.

Text Generator

The Text Generator can be used to generate any text which contains table and/or column names. You specify a "skeleton text" specification – AQT will substitute (and repeat) the table and column names at the places you specify.

This can be used for generating:

- lists of table or column names (for importing into reports or spread-sheets)
- SQL statements or utility statements involving a list of tables
- program source
- HTML source for displaying / editing a table

This tool is accessed by **Tools->Generate Text** from the Table Information window. This tool operates on the contents of the middle and right grids of the Table Information windows so make sure these are populated.

Generate Text is a very powerful tool. Once you become familiar with how it works, it can save you a large amount of time.

The best way of understanding how this works is by an example. In the text window type **Table name is $2** then click on Generate. The following text will be generated:

- Table name is Cust_Orders
- Table name is Customer_Contact
- Table name is Customer_Details
- Table name is Order_Details
- Table name is Order_Line
- Table name is Part_Details

Notice that where you coded $2 the table name has been substituted, and this line has been repeated for every table in the table list.

For other example of the use of this, click on Examples and select one of the items. These will show you the types of things you can generate with this utility (though they may not necessarily be examples of working code).

The substitutional parameters you can specify are as follows. These relate to the values displayed in the grids in the Table Info window:

- $1 Schema name (value in database explorer)
- $2 Table name (value in middle grid)
- $3 Column name (value in right grid)
- $4 - $9 (other columns in right grid – Type, Length, Scale etc)
- $-3 For COBOL users – this is the Column names with underscores replaced with minus signs. This will give you the Cobol host variables names.
- $0 Number of table/column names to be generated.
- $n Sequential number of the table/column name
- $fn A function – see later.

There are a couple of options about the way the text gets generated. The examples demonstrate the use of this. Alternatively you can try them to see what affect they have.

Repeat What?
This option is only relevant when your text consists of more than one line. It specifies what to repeat when you code, say, a $3 in a line – just the line containing the $3, or the entire text.

Delimit Generated Names With
When a parameter is substituted and repeated, what delimiter is used to separate the generated names? Options are:

- newline (each item is placed on a new line)
- comma plus newline. This option can be useful when you are generating a comma-delimited set of names. It is preferable to use this rather than put a comma in your skeleton text; when using this option you won't get a comma after the last item.
- comma. Actually comma plus blank.
- tab. Useful if you are generating a list of names to be pasted into Excel.
- none.

Functions
You can specify functions to do simple string manipulation and conditional processing. This done using the **$fn** keyword. The syntax of this is:
$fn(function, parm1, parm2, parm3, parm4)
The number of parms required depends on the function. Strings do not need to be in quotes.
The following function have been implemented:

$fn(lcase,x)	converts x to lower case
$fn(ucase,x)	converts x to upper case
$fn(scase,x)	converts x to sentence case
$fn(trim,x)	removes leading and trailing blanks
$fn(repl,x,y,z)	replaces all occurrences of string y in x to z
$fn(min,x,i)	returns the minimum of x and i
$fn(max,x,i)	returns the maximum of x and i
$fn(left,x,i)	returns the left-most i characters of x
$fn(right,x,i)	returns the right-most i characters of x
$fn(mid,x,i,j)	returns a substring from x, starting at i, length j
$fn(instr,x,y,z,w)	returns z if x contains string y, otherwise returns w
$fn(ifblank,x,y,z)	returns y if x is blank, otherwise returns z
$fn(ifnum,x,y,z)	returns y if x is a numeric data type, otherwise returns z
$fn(ifeq,x,y,z,w)	returns z if x = y, otherwise returns w

Notes:

- in these examples, i and j are numeric. If a non-numeric value is passed it will be interpreted as zero.
- $ifnum needs some clarification. It doesn't check whether x is numeric, but whether x equals one of: Integer, Smallint, Bigint, Long, Double, Number (eg. a string representing one of the numeric data types). Normally x would be $4.
- if one of your parameters (x, y, z etc) contains either a comma or a bracket then AQT will get

very confused and probably produce the wrong result. Enclose the parameter in single or double quotes.

- if a parameter is omitted, it is assumed to be a null string. For example $fn(repl,$3,_) will remove all _'s from the column name.
- functions can be nested - for example $fn(trim,$fn(lcase,$3))
- additional functions are easily implemented – please let us know if you want any others.

Click on Examples to see examples of how these functions are used.

Save / Retrieve

These options can be used to Save/Retrieve your text generation specification.

Font

This allows you to set the Font in the text box to non-proportional (eg. fixed width) font. This is useful if you want to line up multiple lines (such as when you are generating code for Cobol).

Miscellaneous Notes

Excel Users
You may want to know where your worksheets are. Look under Show->System Tables.

AQT and Decimal Separators

Within Windows it is possible to specify a **Decimal Separator** other than fullstop (you do this with Control Panel->Regional Settings). When this is set to a comma (for instance), numeric values will display as 3,5 rather than 3.5.

AQT behaves in the following way:

- columns defined as FLOAT or REAL are formatted by AQT. In this case the Windows Decimal Separator will be used to format the number.
- columns defined as DECIMAL or NUMBER are formatted by your ODBC driver, rather than AQT. Whether these show the correct Decimal Separator is determined by the capabilities of your ODBC driver. Most databases **do** format the number correctly, however we cannot guarantee this.
- for Oracle, the display of numeric fields is determined by the NLS_NUMERIC_CHARACTERS session variable. You can set this with a statement such as: **alter session set NLS_NUMERIC_CHARACTERS=',.'**

To further complicate matters, most databases do not recognise a comma Decimal Separator in **numeric literals**. In other words, if you are specifying a numeric value in a Where clause, you **must** specify it with a fullstop, irrespective of what you have as your Windows Decimal Seperator. The same applies for numeric values coded in Insert and Update statements.

AQT helps you use the correct format for a numeric literals; when you use **List Values** to display the values of a numeric column, AQT will display it using a fullstop instead of a comma. Similarly, when you go into Update Mode, AQT will automatically change all your numeric values to have a fullstop. These features will ensure that your numeric literals will be specified in the correct format.

Connection Problems
If you are unable to get past the signon window then you are having problems with the database connection. If you are lucky you might get an informative error message, however this is often not the case.

There are many things that can cause connection problems, most if them outside the control of AQT:

- ODBC datasource not configured correctly or database does not exist.
- Database is not started or cannot be accessed due to a communications problem.
- Userid and/or password not valid.

Signing onto a database is a surprisingly complicated subject – see Database Signon for a fuller explanation of this.

Try Connecting from ODBC Setup or another Product
This will tell you whether it is a problem with the ODBC connection, or a problem with AQT.

- many ODBC Setup drivers have a "Test Connection" option which is very useful (get into this from Control Panel->ODBC).
- alternatively, try accessing you database from a product such as MS Access (File->Get External Data->Link Tables. Select Files of type ->ODBC Databases. Select tab Machine Data Source then select your database. Geez what a lot of work just to sign onto a database).

Prompted Signon
One thing you can try is a Prompted Signon. Select this by setting **More Connect Options** and check **Prompted Signon**. In this mode the ODBC driver will prompt you for connection information. This gives you better control of seeing and specifying any particular connection parameters.

Prompt for Missing Info
This is like **Prompted Signon** but only prompts you for the those connection parameters which are not currently present. In many cases this is a more suitable option than **Prompted Signon**.

Userid has No Password
If your userid has no password you can't just leave the password field blank; AQT will assume you are doing an "integrated logon" and specify neither the userid or password in the signon request. In this case you need to select **More Connect Options** and check **No Password**.

"Error Requesting Parameters"
You may get a message such as "error requesting parameters required to complete a connection". If you get this then you haven't specified enough information in your ODBC configuration:

- for MS SQL Server and Sybase you can get this if you have specified the name of the machine to connect to, but not the database name (eg. master).
- for MS Access you get this if you did not specify the name of the mdb file.

In both cases, go into Control Panel->ODBC, select the database and hit Configure. Correct the settings.
Alternatively you can specify the Database when you log on. Select **More Connect Options** and fill out the **Database Name** box.

IM026
DB2 users will get this if their c:\sqllib\bin directory has not been set in their PATH. The DB2 client install (for some releases) fails to do this.

System Hangs with message "Loading Schema Information"
If you are a DB2 user and you get this, the probable cause is that the "ODBC packages have not been bound against the database". This may not mean much to you, but it should to your DBA. If you are the DBA then you can bind these by a) going into the Client Configuration b) selecting the database c) click on Bind d) select Bind Db2 Utilities.

Unable to Display Table Information

This section applies if you gotten past the signon window, however the system fails to correctly show tables and the information about them.

If the system is getting errors while trying to get this information, they will be written to the status bar at the bottom of the window (if the message is too big to fit into the status bar, clicking on it will display it in full). These may shed a light on the problem.

There are two things you can do to try and fix the problem.

Switch off System-Table Queries

The problem could be due to errors while processing the system queries. These can be switched off by:

- logging off the database (back to the Signon window)
- selecting **Options->General Options->Table Info**. Select **Use ODBC**.
- log back onto the database.

This will cause AQT to operate in a more basic mode. This works OK with all databases we have tried it with. You can still access tables and run queries. The only things which will be absent are the system-table queries and some of the DBA tools.

Switch off Parameters

You might try this instead of the previous item.
If you are getting errors while processing the system queries this could be due to the use of parameters. A number of databases have bugs in this area.

- select **Options->General Options->Parameter Usage**. Select **Use Substitution**.

(this comes into effect immediately; you do not need to log off/on).

For a technical discussion on this subject see Parameter Usage.

Known Problems

The following known problems exist with AQT. A number of these are due to peculiarities/bugs of ODBC drivers and/or the databases. You may or may not experience these problems depending on what level of software drivers you are using.

Area	Problem Description
Column Types for Sybase Enterprise and MS SQL Server	These are held in the system tables in a very complex way. Furthermore we have developed our queries to work on both SQL Server 6.5 and 7.0; this adds an extra degree of difficulty. Hopefully our queries will report on this information accurately but no guarantees!
Column Types for MS SQL Server	The column width is reported inaccurately for nchar and nvarchar columns (these are stored as Unicode and occupy double the space to what has been declared – AQT picks up the space used). This will be fixed soon.
Parameter Usage with Sybase Enterprise	This doesn't seem to handle numeric parameters. Circumvention: use **Options->Parameter Usage->Use Substitution**, or code your query as **where id=convert(int,?)**
Cancelling Queries	With DB2 this can cause a problem. While you can do this (and it works), it seems to muck up the cursor and prevents further queries from running. Looks like a bug in DB2 (and I have the system traces to prove it!). Ditto for query timeout. This problem can be circumvented by (in the DB2 Client Configuration) setting Cursor Hold to No.
Sorting Grids	This doesn't work very well with date / time values. A VB peculiarity.
Default values in DB2/390	These are held in a complex way in the database that AQT doesn't yet handle accurately. This also affects Default Values in the DDL generation of DB2/390 tables.
Object Not Set	Occasionally AQT starts to fail with the message "Object Not Set". This is an intermittent problem that we've never gotten to the bottom of. It seems to only happen in a memory-constrained machine. Circumvention: sign off and back on to the database.
Automation Error	Don't you just love these informative error messages? Well-done Microsoft! Again, an intermittent problem with the ODBC drivers. Solution unknown.
MS Access gets "Statement not Valid at this time" when running update.	Switch off Safe Update Mode (see Options->Statement Options).
AQT sometimes remains active after exiting.	We've never understood this, despite spending a lot of time on this problem. Circumvention: cancel the task using the Task Manager.

ODBC Table Info

Many problems exist if you use **Options->Table Info->Use ODBC**. You wouldn't normally do this. When running in this mode AQT gets information on the tables from the ODBC API calls, as opposed to querying the system tables. There appear to be many bugs with these APIs.
Some of these problems also exist for non-catalog databases such as Access.
The circumvention for all these problems is to use Options->Table Info->Use System Tables.

- Oracle Version 7 gives duplicate values for the table owner names.
- Oracle and DB2 are sometimes confused about what is a Table and what is a System Table.
- Most databases don't give anything for the column Default value.
- DB2 doesn't give anything for the column Description.

SQL Statement History
This window is displayed from the Run SQL window. Click on the History icon, **Windows->History** or Ctrl-H.

This shows the last 100 SQL statements you have run from the Run SQL window. The following information is shown:

- **time** the statement was run
- the **text** of the SQL statement
- **elapsed** time for the query
- whether the query results were **saved** to disk
- the **result** message of the query. This will generally be either an error message (if the statement failed) or a message giving the number of rows retrieved.

If you click on a row you will be shown full details of the query at the bottom part of the window – the full text of the query and / or error message. Clicking on **Retrieve SQL** will retrieve the SQL to the Run SQL window. To exit without retrieving anything, click on the **Close** menu item.

If your **Query Results** were saved in the Query Display then a little table-icon will be shown in the **Saved** column. If you click on this you will immediately be shown the query results.

The SQL Statement history is saved to disk between AQT sessions. The file it is saved to is **history.aqt** in the aqt install directory. The history is saved here whenever you sign off a database (eg. exit the Table Information window). However you can "manually" save it by clicking on the **Save** option of the History window. You might do this if you have run some queries you want to save, and you don't want to risk losing them if aqt crashes.

GUI Query Builder

The Gui Query Builder is a new feature in AQT V4. It can be accessed by:

- from the Table Info window, click on a table them select the **Build Query** menu item
- from the Run SQL window, click on the Query icon or **Build Query** menu item
- in the Table Info window, select a view, right-click then select **View as Query**

The GUI Query has the following major features:

- allows you to **build a query** using the GUI paradigm – eg. a graphical display of tables. This is particularly useful for building joins between tables and for easily visualising a query.
- most aspects of a query can be specified – column order, column functions, column aliases, constants, summary queries (group by), row order, where clauses.
- AQT can also **reverse-engineer a query**. It can take an existing SQL statement and turn it into a graphical query. This can be SQL generated by AQT, or some existing SQL you have. You can amend this or reformat the SQL using the many options AQT provides for the format of the generated SQL.
- AQT can also build and reverse-engineer **View definitions**.
- There are limitations to the SQL that AQT can build or reverse-engineer. Click here to see a list of these.

Joining Tables

- Joins between tables can be quickly defined by dragging between the related columns in the GUI.
- When you have a "known" relationship between your tables (eg. it is defined to your database as a foreign key relationship) AQT can automatically build a join between these tables with a single click.
- There are many cases when you have a relationship between your tables, but this is not defined to your database. In this case AQT allows you to save the definition of the relationship as a user-defined relationship. The details of the relationship are permanently saved (to disk); you can then build a join between these tables with a single click.

Starting the GUI

When you start the GUI you will be shown a window with the following tabs:

- GUI – used for showing a graphical display of your query tables and joins.
- Tables / Joins – used for specifying further information about the query tables and joins.
- Columns – used for specifying further information about the columns in the query, and building complex query-columns.
- Where – used for building the WHERE statement for the query
- Other – miscellaneous other options for the query. If the query is a view definition this will show you the View name plus other options related to the View.
- Options – use this to specify the many options you have for the way the SQL is generated.

In addition, at the bottom of the window is a text box that has the query SQL. This will change as the query is built or amended.

Menu Icons

There are the following icons at the top of the screen:

- Tick. **Save query** – return the generated SQL to the Run SQL window, from where the query can be run.
- Cross. **Exit without saving** the query.
- **New Table**. Add table to query. This is described in GUI Tab.
- **Related Tables**. Add all related tables to the query. This is described in GUI Tab.
- **Key**. This will get the Primary Key information for all the tables in the query. By default AQT will not do this (AQT will not waste your time getting information you may not want).
- **Refresh Query**. This button is used to return the query to the way it was when you entered this window. All subsequent changes are lost.
- **Refresh Table Info**. For performance reasons, AQT caches in memory information about the tables in your query. This includes the columns, primary keys and foreign keys of the tables in your query. This reduces the number of times AQT has to query the system tables, which can sometimes be a slow process. However there is a disadvantage to this – if you change a table (eg. add some columns, or add a foreign key) then the query builder will not show this change. It will continue to use the old definition. If this happens, click on the **Refresh Table Info** icon. The information on the tables is refreshed from the system tables.
- **Query Assistant On/Off**. This switches the Query Assistant on or off.

Gui Query Builder - GUI Tag
This window is seen by clicking on the **GUI** tab of the GUI Query Builder. Refer to this topic if you want general information on the features of the Query Builder.

The GUI tab shows you:
- the **tables** which are in your query. You can move these around, resized etc.
- the **joins** between the tables.

Notes:
- the **primary keys** of the tables will be shown in blue. You will need to click on the Key icon to see these.
- sometimes you might see a completely empty table-box. This happens when AQT is unable to find the columns for the table.
- to understand what the **menu icons** do, see GUI Query Builder.

Note on a Strange Bug
There is a peculiar bug in AQT and/or Windows. With this, the tables jump a few centimetres when you click on them. This problem can be prevented by changing a Windows setting. Go into Control Panel->Display and find the option Show Window Contents While Dragging (sometimes in the Effects tab). Have this option off.

Query Assistant
This feature is new to V4.1.3. We actually developed this for V4.1.0 but decided we didn't like it and removed it. Please give us some feedback as to whether it is a useful feature or not.
The Query Assistant is activated by clicking on the Query Assistant icon. It gives you various hints and information about how to use the query builder – try it and see what it does.

Adding Tables to the Query
You can add tables to your query in two ways:
- click on the **Add New Table** icon
- click on the **Add All Related Tables** icon

Add New Table takes you into a dialog where you can specify a table to be added to your query. It also allows you to see all tables which are related to you current query-tables.

Add All Related Tables will automatically add to the query all the tables which are related to the currently-selected table. Before doing this, click on one of the tables to select it (you will get a message at the bottom of the screen saying which table is selected). Other notes:
- for this option to work, your table must have some "known relationships" – either defined to the database (as primary/foreign keys) or set up as user-defined relationships.
- this option can take a few seconds to run. AQT queries the system tables to get the information on related tables. This seems to be particularly slow for Oracle.

Joining Tables
If you use **Add New Table -> Add Known Relationship**, or **Add All Related Tables**, then your tables will be joined automatically by AQT.
To join tables manually:

- click on the column you are joining **from**. You can only select a single column. The column will be highlighted in yellow. Information about the column will appear in the Status Bar at the bottom of the window.
- holding the mouse button down, drag to the column you are joining **to**. As you do so, the message in the Status Bar will tell you about the join you are about to create. Pay particular note to the data-types. In general these should be the same for both join columns. Release the mouse-button to create the join. Click on the background to not create the join.

Notes:
- generally, you should only join columns which are of the same type and length. If you try to join columns which are of different type, AQT will give you a warning message. You will have the option of aborting the join (in case you have made a mistake).
- if your join is a multi-column join, you will need to repeat the join procedure for every column in the join. As you do so, you will get the message "This relationship has been added to the existing join". You only get one join-line between the tables, even though there are multiple columns in the join. This is in contrast with some other products, which show one join-line for every pair of columns in the join (in our opinion this can make the query GUI very cluttered).

Other notes:
- you cannot join a table to itself. If a table has a relationship with itself (which is quite a valid thing), you need to add the table twice to the query and join these. Even though these two tables have the same name, they will have different table-id/correlation names, so can be referred to without ambiguity.
- similarly, there can only be one relationship between two tables. If TableA has two relationships to TableB (again, this is quite normal) you must create two instances of TableB in the query and join TableA to each of these.

Viewing or Amending Join Information

If you click on a join-line, the columns which are in the join will be highlighted. This is a very quick way of seeing the definition of join.

For other options for a join, right-click the join. You will shown a pop-up menu with options:
- **Join Details**. Takes you into the Modify Join window where you can view or modify the details of the join.
- **Delete Join**.
- **Remember**. This will save this join as a user-defined relationship.

Removing a Table from the Query

You can do this by clicking on the Close (x) button on the table-window.

Selecting which Columns are in the Query

In the table-windows, the table-columns are shown and whether the column is in the query. You can include or exclude the column from the query by checking / un-checking the column check box.

The table-windows have menu options **All** and **None** to quickly select All or None of the columns for the query.

There are many other attributes about the query column which cannot be specified here, for

instance the order in which the column appears in the query. To specify this and other options, click on the Columns tab.

Gui Query Builder - Table / Joins Tag
This window is seen by clicking on the **Table/Joins** tab of the GUI Query Builder. Refer to this if you want general information on the features of the Query Builder.

The information displayed on this screen is very similar to those displayed on the Gui screen. However this allows you to specify these options in a non-GUI way, for those people who prefer this style of interface.

The one option which can be specified here, and nowhere else, is the Table-Id (described below).

This window shows:

- the **tables** which are in the query. You can add a table to the query with **New Table** and remove one with **Delete**.
- the **joins** between the tables. You can add, modify, delete joins with the **New, Modify** and **Delete** buttons.

New Table
This operates slightly differently in this window than it does in the GUI tab. After adding a new table to the query, you will be asked to specify a join for this table. This is save having to go into New Join after New Table.

Table-ID
Tables can have a Table-Id, which is a short abbreviation for the table name. The correct name for table-id is the "correlation name". Use of a short table-id can make the SQL less voluminous and easier to read.
If a table appears twice in the query, it is essential to have a table-id to distinguish which of the two instances of the table you are referring to.
AQT will by default generate a table-id for the tables. To change this, click in the Table-Id column and enter/amend the value. Then hit Enter or click elsewhere to effect the change.

Gui Query Builder – Columns Tag

This window is seen by clicking on the **Column** tab of the GUI Query Builder. Refer to this if you want general information on the features of the Query Builder.

The window shows the all the query columns. It allows you to:

- specify which table columns are in the query
- specify additional options about the query columns
- build more complex query-columns, such as constants, sub-selects etc
- specify the order in which the columns appear in the query
- specify the sort-order for the query (Order-By)
- whether the query is a standard query, or a summary query (Group By)

Query Columns versus Table Columns

To clarify some of the terminology we use here:

- **table-column** – this is a column in one of the tables which is in the query
- **query-column** – this is a column which is in your query.

This is an important distinction to make, as query columns and table-columns can be quite different:

- a query column can be a combination of table-columns, and functions on table-columns. Examples: substr(col1,1,20), col1*100/col2.
- a table column can appear in more than one query column.
- a query-column can not be related to any table-column. For instance it could be a literal, a system-variable or a sub-select. We sometimes refer to these as "non column-based" query columns. Examples: NULL, sysdate(), 'Name', 23.

AQT allows all these type of query-columns to be built.

Table / Columns

On the left is the list of tables and columns. You can use this to specify which columns are to be in the query. You can include all columns in the query by clicking on **All**. You can exclude all columns by clicking on **None**.

Query-Column Grid

On the right is a grid showing the columns in the query. This has:

- **Table / Column**. This identifies the table-column the query-column comes from. These fields will be blank for a non-column-based query column.
- **Function**. Gives you column function (if any). To add a column function, click in this cell. You will be shown a drop-down list of the common functions for that column type. These functions will shown as, for instance, **substr(:,1,20)** – the column name will be substituted where the colon appears. Select a function from the list, or type one in. If you need more space to enter a complex column function, click on the **Mod** button to enter the Query Column window.
- **Display Name**. This gives the name of the Query Column. When you run a query, your database (Oracle etc) will give a name to all your query columns. You see this in the Query Results display. By default, the Query Column name is the same as the table-column name. In some circumstances, your database will give the columns a system-generated name. If you wish to specify a particular name, click in the Display Name cell and enter the name.
- **Order By**. This specifies whether the database is to sort by this column. If you click on this cell you will get (for example) "1+" displayed. This means the column is the first in the sort

order, and it is to sort Ascending. Click again and it will change to "1-" (sort descending). Click again and you will get "" (no sort). You can specify up to 10 sort-columns.

- **Group By**. You will get a Group By column in the grid if you have specified a Summary Query. Click on this to specify the column as being a Group By column. For a further discussion on this click here.

New
Use this button to add a new query column. You might use this option to add a non-column-based query column. Another circumstance when you might need to use this is if one of your columns appears twice in you query (for instance "Select name, left(name,1,5) from table") (this is the only way you can do this). When you select this option you will be taken to the Query Column window.

Mod
Use this button to modify a query column. This takes you to the Query Column window from where you are given full details of the query column. You can modify these as necessary.

Del
This deletes a query column. If you have selected a range of column, all of them will be deleted.

Up/Down
Use these buttons to change the order in which your columns appear in the query.

Distinct
If you select this option, your query will only display distinct rows. Duplicate rows will be removed. This option is generally of not much use if you are displaying all columns in the table; it is often used to display the different values a column can take. In this case you will specify only one or two columns in the query.

Query Type
Specifies whether you are running a **Normal Query** or a **Summary Query**. A Summary Query is used to summarise the contents of the table. When you specify a Summary Query another column (Group By) will appear in the column grid. You use this to specify which columns you are summarising by. Click here for a description of summary queries.

Summary Queries

Summary queries (also called **Group-By** queries) are used to summarise the contents of a table.

There is an important rule about summary queries which you need to be aware of. **All** columns in the group-by query need to be either:

- a column you are **summarising by**
- a column you are **summarising**

This is best illustrated with an example. The following query summarises customer revenue by customer and month:

Select customer_code, month, sum(revenue) from orders group by customer_code, month

In this:

- customer_code and month are the columns you are **summarising by**
- revenue is the column you are **summarising**.

The columns you are summarising-by are specified as **Group-By** columns. In the Query-Column Grid you select these by clicking on the Group-By column.

The columns you are summarising-by must have a **summary-function**. This indicates **how** you are summarising the column. In the above example, the summary function is sum(). Other functions are: min(), max(), avg(). These can be seen in the Functions drop-down list.

So, the rule for a summary query is that **all** columns must either:

- have the Group-by column clicked
- have a summary column-function specified

When you are doing a summary query you can also specify a HAVING to select which summary rows are in your result-set.

Having Clause

When you are doing a summary query, you can also specify a Having clause. This is like a Where clause, except that it involves a summary value, rather than a column value.

In our example of customer revenue, the following query shows you customer revenue when it is more than $1000:

Select customer_code, month, sum(revenue) from orders group by customer_code, month having sum(revenue) > 1000

When you have a summary query, the Where tab and Where Clause windows include options for dealing with Having Clauses

Having versus Where

The easiest way of understanding the difference between a Where clause and a Having clause is as follows.

- A WHERE clause specifies which rows in your table(s) are to be included into the summarisation process.
- The HAVING clause operates after the summarisation has been done. It specifies which of the summary-rows are to be returned in your result-set.

Gui Builder – Where Tab
This window is seen by clicking on the **Column** tab of the GUI Query Builder. Refer to this if you want general information on the features of the Query Builder.

This window is used for building a WHERE statement for a query. The Where statement is used to specify which rows are returned by the query.

A Where statement consists of a number of Where clauses. For instance the following Where statement:
(name='fred' and state='CA') or (salary<10000 and name is null)
consists of 4 Where Clauses. These clauses are connected by ANDs and ORs; and indented within various levels of brackets.

The Where window shows you the following:
- the grid shows you your Where clauses, and how they are connected
- underneath this is a text box which shows you the entire Where statement.

You build your Where statement in two steps:
- build your Where clauses
- specify how the Where clauses are connected

These two steps are described below.

Building your Where Clauses
You do this by clicking on **NEW** (build a new clause), or **MOD** (modify an existing clause). **DEL** deletes a Where Clause. NEW and MOD will take you into the Where Clause window where you can build a Where Clause.

Specifying how the Where Clauses are Connected
This is done using the Grid within the Where window. Before doing this you will need to have built some Where clauses – these will be displayed in the grid.
- by default, the Operator connecting your where clauses is AND. Clicking on the Operator column will toggle this between AND and OR.
- you can add brackets before and after the where clause by clicking in the (and) columns. Left-click to increase the bracket level. Right-click to decrease the bracket level.
- you can move the Where clauses up and down in the list with the **UP** and **DOWN** buttons. This will not change any of the operators or bracketing levels.

Displaying the Where Statement
As you build the Where statement, it is shown in the text box. If you have a very large or complex Where statement you might find it useful to check **Show Clauses as Numbers**. This shows clause numbers in place of clause text. For instance **(name='fred' and state='CA') or (salary<10000 and name is null)** will be shown as **(1 and 2) or (3 and 4).** This can make it easier to understand complex Where statements.

Show What
When you have a summary query, you can specify both Where clauses and Having Clauses. The **Show**

What option allows you to specify which of these statements you display in the text box.

Limitations of the Gui Query Builder

SQL is a very powerful language: to develop a query-builder which can deal with all SQL constructs would be a very big job. Below are a list of the limitations of the AQT query builder.

- sub-selects cannot be built or interpreted, either in the Column list, Table list or Where clause. This is perhaps the biggest limitations which we will address in later releases.
- when interpreting a query, AQT requires all keywords to have a blank on either side. Some databases are smarter than AQT and are happy with constructs such as WHEREA=B. AQT will not find this WHERE clause and, as a result, will get quite confused.
- AQT cannot build or interpret the WITH clause, and recursive queries.
- Column synonyms need to be specified with the AS clause "Select col1 AS fred from tab1" OR (for SQL Server / Sybase) "Select fred=col1 from tab1". AQT cannot interpret a column synonym with the syntax "Select col1 fred from tab1".
- Unqualified table names sometimes causes confusion. When AQT finds an unqualified table, it needs to know the Schema name in order to get the column information. First it tries a Schema of the signed-on userid. Then (for Oracle) PUBLIC, or for Sql Server/ Sybase, dbo. Then it tries to look for an alias.
- AQT can get confused if you have tables with the same name, but different case. Eg. if you have tables EMPLOYEE and "Employee" AQT can get the wrong one.
- AQT cannot deal with tables in another database (eg. a three-part table name such as master.dbo.systypes).
- Joins can involve a maximum of 20 tables.
- Outer Joins are a very complex area, especially when you have more than two tables involved. For instance, in a three-table query the following possibilities exist:
    - TabA OUTER JOIN TabB ON taba.col1=tabb.col1 OUTER JOIN TabC ON taba.col1=tabc.col1
    - (TabA OUTER JOIN TabB ON taba.col1=tabb.col1) as x OUTER JOIN TabC ON x.col1=tabc.col1
    - TabA OUTER JOIN (TabB OUTER JOIN TabC ON taba.col1=tabc.col1) as x ON taba.col1=x.col1

These three queries all give different results. With more than three tables, it gets even more complicated. It's all too much for my poor little brain to cope with. I need an upgrade. AQT only deals with format 1. Given sufficient user demand we may extend AQT's capabilities to deal with these other join types.

Build Where Clause
This Window is used to build an individual Where Clause. You enter this window by clicking on **NEW** or **MOD** in the Gui Where tab. Refer to this if you want general information on the features of the Query Builder.

The layout of the window reflects the normal format of a Where clause, which is:
function(column_name) operator value
Examples:
substr(cust_name,1,2) = 'FR'
cust_type NOT IN ('a','b')
purchase_time > CURRENT TIME – 1 DAY

In this:

- **Table/Column** specifies the column the Where clause is based on. You first select the table; when you do so AQT will show you all the columns in the table. If you select **Show->Query Cols**, AQT will only show you those columns which are in the query. Select the appropriate column.
- **Column Function** specifies an (optional) column function. You specify this in the format **function(:)** – the column name will be substituted wherever the colon appears. In the listbox you will be shown a list of common functions for the type of your column. You can select one of the or type one in. If you have a long column function, click on the **>>>** button to get an expanded window where a long value can be entered.
- **Operator** specifies the where-clause operator. The listbox has most of the common where clauses. For some clauses (**In, Between, Like**) you can qualifier the operator with a NOT. You do this by checking the **NOT** checkbox. Note that for most databases **not equals** is given by **<>**.
- **Column Value** specifies the value the column is compared to. There are a number of ways you can specify this: type the value into the box, or click on **>>>** to get an extended box for longer values. **Get Values** will show you all the existing values of the column. **Get Constants** will show you common system variables (and other constants) for the column.

As you build the where clause, the full text of the clause will be displayed in the **Where Clause** text box.

Quotes
This specifies whether the value in the Column Value box is to have quotes placed around it. AQT will (normally) be smart enough to set this appropriately (eg. will be set for character values, not set for numeric values). In some cases you may want to set this yourself.

BETWEEN Clause
When you have a BETWEEN clause you specify **two** values for the column (eg. BETWEEN 1 AND 10). AQT allows you to specify these two values independently; you will get a box **For BETWEEN this is** where you can specify whether the value you are specifying is the first or second value on the Between Clause.

IN Clause
When you have an IN clause you can specify **multiple** values for a column - eg. IN ('A','B','C','D'). There can be many of these values; some databases allow you to specify several hundred. In future

releases of AQT we may build a facility for easily manipulating large IN clauses. In the current release, you can specify multiple IN clause values by selecting multiple items from the value list (using the Ctrl and Shift keys).

Sub-Selects

Sub-selects can be used in a Where clause, normally in conjunction with the IN operator. Example: **col1 IN (select distinct col1 from tab2)**. AQT allows you to enter a sub-select (eg. you can type it into the **Column Value** box), however it gives you no assistance in building it. This will be done in a later release of AQT.

Clause is Column Based?

The above discussion relates to Where clauses which involves comparing a column to a value. We call these "Column-based Where clauses". The vast majority of your Where Clauses will be like this. However it is possible to have Where clauses which are not in this format. One example of this is a clause such as:

Exists (Select * from Tab1 Where Tab1.ColA = Tab2.ColB)

You can specify clauses such as this by setting **Clause is Column Based** to **No**. All the upper windows will be greyed out – you type the where clause directly into the Where Clause text box.

Having Clause

If you are building a summary query, you have the option of building a Having Clause as well as a Where Clause. In this case this window will have an option box **Having Clause** (yes/no). Use this to select whether you are building a Where Clause or a Having Clause. If building a Having Clause, the Column Functions will show column summary functions. For a Having Clause you must select one of these.

Gui Builder – Other Tab

This window is seen by clicking on the **Other** tab of the GUI Query Builder. Refer to this if you want general information on the features of the Query Builder.

This window is used to specify miscellaneous options related to the query. At this stage it is used to specify the Statement Type. AQT can be used to build more than just Select statements. It can also build:

- View Definitions
- Select INTO statements (Sybase only)
- Any other SQL statement that contains a Select statement. This is explained further on in this section.

If you have built a Query, you can change your query to one of these other types. Change the Statement Type then enter one of the options relevant for that type:

- For **Select into** statements (which creates a temporary table), enter the name of the temp table
- For **Create View** statements, enter the name of the View. When creating a view there are two styles of create-view statements – with or without an explicit column list specified. Select and de-select this option to see how this affects the generated SQL.
- For **Other** statements, specify the "wrapper" for the query. The wrapper if the SQL statement which contains your query. When coding this wrapper text, specifying **$sql** where the query-sql is to be placed. An example of this is below.

Other statements need more explanation. Suppose you have built a query, and you wish to insert the query-results into another table. Change the Statement Type to Other and code a wrapper of:
Insert into Customer_Details $sql
The query you have built gets substituted where you have coded $sql in this.

Minor Hassle

If you change some of these options you often have to click elsewhere for your changes to be picked up – click in the SQL text at the bottom. The SQL gets regenerated on the "lost focus" event on the fields you are changing.

Gui Builder – Options Tag

This window is seen by clicking on the **Options** tab of the GUI Query Builder. Refer to section if you want general information on the features of the Query Builder.

This window has a large number of options which specify how the SQL statement is formatted.

Most of these are obvious and are not described in detail. You can see what they do by selecting them and seeing the effect on your SQL.

Some options which may not be obvious and need to be explained are:

Select * When Appropriate. By default, AQT will generate the SQL as "Select * from table" when this is appropriate. It is appropriate when a) the query includes all the table columns b) the columns are in the same order they are in the table c) no column functions or correlation names are used. If you de-select this option, AQT will include all the columns, instead of an *.

Always Qualify Table Names. This is a slightly obscure option. It applies when AQT has reverse-engineered a query which uses unqualified table names (eg. table names which do not specify the Schema). In the generated SQL, AQT will use the table-name as given (eg. unqualified). Select this option if you want AQT to include the table qualifier.

Ignore Case in Correlation Names. This applies to the Display Names you set up for columns. It governs how AQT handles a display-name which has lower-case characters – if you want the case preserved you must de-select this option. For instance, suppose you give your Customer column a display name of **Cust**. With this option selected (the default), AQT will generate the column clause as **CUSTOMER as Cust**. Some databases ignore the case in the correlation name and will display it as CUST. If you want the case used, de-select this option. AQT will then generate the column clause as **CUSTOMER as "Cust"**. The database will display the column as Cust.

Names in Order By and Group By

When you have an Order By and Group By clauses, there are a number of options about how these are specified in the Order By and Group By clauses. This is best explained with an example. Take the following query which you wish to order by the first column:

Select Customer as Cust, Customer_Name as CustName from Customer_Details

- With **Column Names** specified you will get: **Order By Customer**
- With **Column Numbers** specified you will get: **Order By 1**
- With **Column Aliases** specified you will get: **Order by Cust**

Note that some of these options may not be valid for your database. For instance, many databases do not allow you to use Column Aliases in the Order By and Group By statements. AQT will not enforce these rules (as we don't want to have to change AQT every time a new release of a database comes out).

Query Column
This Window is used to build or modify an individual Query Column. You enter this window by clicking on **NEW** or **MOD** in the Gui Columns tab.

The layout of the window reflects the normal format of a query Column which is:
function(column_name) As "Display Name"
Examples:
cust_type
substr(cust_name,1,2) As "Cust Prefix"

Table/Column
Specifies the column the Query Column is based on. If you are modifying a Query Column these boxes will be disabled.

Column Function
Specifies an (optional) column function. You specify this in the format **function(:)** – the column name will be substituted wherever the colon appears. In the listbox you will be shown a list of common functions for the type of your column. You can select one of these or type one in. If you have a long column function, click on the **>>>** button to get an expanded window where a long value can be entered.

As you build the query column, the full text of it will be displayed in the **Column Expression** text box.

Column Based?
The above discussion relates to Query Columns which involves a column from one of your query tables. We call these "Column-based Query Columns". The majority of your Query Columns will be like this. However it is possible to have Query Columns that are not based on a Table Column. Examples of this:
Literal – eg. 'Test' , 0
System Constant – eg. Sysdate(), Current_Date
Sub-select – eg . (Select count(*) from Tab1)
You can specify clauses such as this by setting **Clause is Column Based** to **No**. The Table/Column window will be greyed out – you type the query column directly into the **Column Expression** text box. In addition, a list of common System Constants will also be displayed in the Expressions text box – you can select one of these.

Sub-Selects
Sub-selects can be used in a Query Column. AQT allows you to enter a sub-select (by typing one into the **Column Value** box), however it gives you no assistance in building it. This will be done in a later release of AQT.

Other Options
These specify a number of other options for the Query Column.
Display Name – gives the name for the Query Column.
Group-By Seq – for a summary query, this specifies that you are summarising by this column (and

the order in the group-by list).
Order-By Seq – specifies to Sort the query results by this column. The **Asc/Desc** specifies whether to Sort Ascending or Sort Descending.
Include Col – for some databases it is possible to Sort by a column, yet not have it in the Result-set. This option allows you to do this; if you are sorting by this column the Include Col checkbox will be enabled. If you de-selecting this, the column will not be included in the result-set.

User-Defined Relationships

The Gui Query Builder can automatically build joins between tables, if the tables are "known" to be related. AQT will "know" the tables are related if the relationship is defined to the database as a foreign-key relationship.

There are a number of circumstances where tables are related, yet this relationship is not defined to the database:

- Your system designers have chosen not to implement the relationship (for performance or other reasons).
- The relationship is between system tables (foreign-key relationships are not defined between these).
- The database does not support foreign key relationships (MySQL is an example of this).
- You are joining views or synonyms – foreign key relationships cannot be defined between these objects.

In these cases you can define the relationship as a "User Defined Relationship". Once you have done this, the relationship between these tables becomes a "known" relationship. You will be able to join these tables with a single click. This will save you a lot of time if you are frequently building queries between these tables.

To Define a UDR

Join the tables as normal (by dragging between columns). Once you have defined the join correctly, right click on the join and select the **Remember Join** option. Alternatively you can edit the join details and click on the **Remember** button. You will get the message "Join Details Saved".

View and Edit UDRs

You can see what UDRs you have by the following:

- In the GUI Query Builder click on the **Add Another Table** icon.
- Click on the **Select Known Relationships** tab. This will show you all the known-relationships involving the current query tables (excluding those already in the query). This will show both database-defined-relationships, as well as user-defined-relationships – the Source column (on the right) shows you where the relationship comes from.
- Click on the **Show User-Defined Relationships** button. This will take you to the **User Defined Relationships** Window.
- From this window you can Edit or Delete the UDR entries.

UDR File Location

The information on the UDRs is held in a file. There is a separate UDR file per database. The UDR file name is database.udr (for instance aqtdemo.udr). The directory where the UDR files are stored is the config file directory.

You can manipulate these files as with any other files. For instance, if you have set one up you can copy it to another persons PC – they will then be able to use the udr definitions you have set up. Similarly, if you have multiple similar databases (for instance development and production). You can set up the udrs for one of the databases, then copy the contents into udr files for the other databases.