Edit-DB: User Guide

Author : Nigel Maddocks
Company : Edit-DB
Version : 2.05.016
Last Modified : Date: 2007-01-07

Table Of Contents

Edit-DB

Edit-DB ver 2.05.016, Doc ver 2.05.016

---

1. Overview

Edit-DB provides a web browser interface to easily view and edit database data, and view an edit other standard database objects for Oracle, MS SQL Server, MySQL, Postgre, IBM DB2, IBM Series i (iSeries, AS/400) and HSQLDB databases.

Edit-DB is fully compatible with Internet Explorer and Firefox web browsers.

Edit-DB is hosted within a servlet container such as Apache Tomcat or Resin. Installation of Edit-DB on the server is simply to drop the application *.war* file into Apache Tomcat's or Resin's *webapps* directory.

Giving a user access to Edit-DB is as simple as informing them of the url, by default *http://yourserver:8080/EditDB*, and providing them with the name of the database server and a userid and password.

Features

- Selection of a database object, e.g. a table, from a list, to display it.
- A *multi-record* display of Table or View data allowing easy selection and sorting of the data without the need for SQL knowledge.
- Display and Print the data
- Display associated data in foreign key tables
- Edit data in-line
- A *single-record* display of a Table or View record, simply by double clicking on a row in the *multi-record* display.
- Display and Print the record
- Edit the record, or delete it or copy it to another record.
- Copy a table, or a selection of its rows, to another table.
- List, Display, Create, Edit, Delete functions for other database object types such as:
- Views
- Functions
- Stored Procedures
- Triggers
- Packages and their objects (Oracle only)
- Indexes (list & display only)
- Sequences (list & display only)

- List relationships between tables and other tables, with the ability to display the relationships graphically using Scalable Vector Graphics (SVG).
- A SQL entry interface to directly enter SQL commands.
- A multi-window option to enable different tables' data, or different objects to be displayed simultaneously.

The browser window is divided into sections, known as frames as in the following diagram which can be compared with the screenshots which follow:

<table>
<tr><td colspan="2">1. database, owner, object-type choice</td></tr>
<tr><td>2. list</td><td>3. object details</td></tr>
</table>

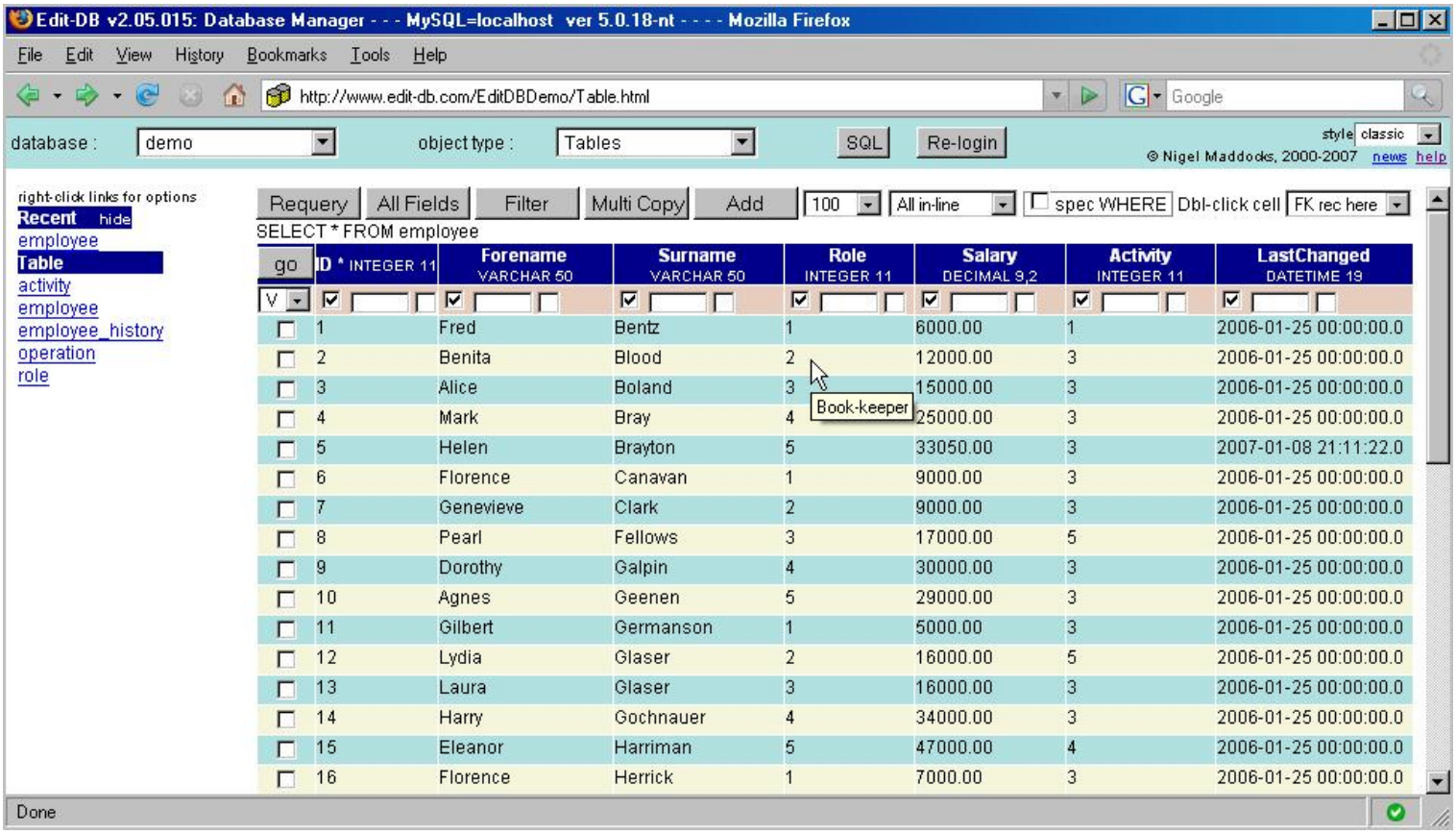
These frames will be referred to as:

1. The Object-Type frame
2. The Object-List frame
3. The Object-Detail frame

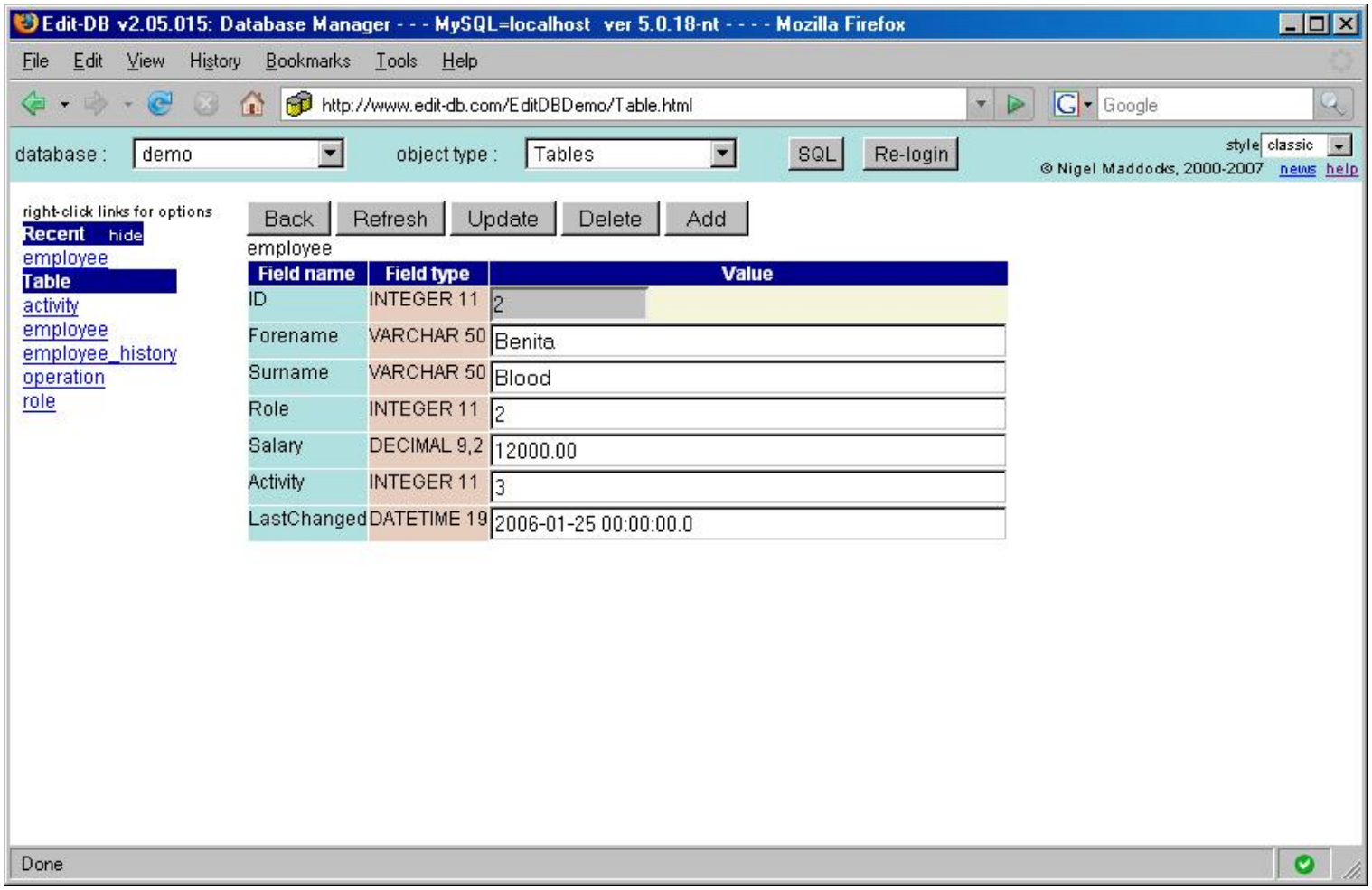
1.1.Screenshots

1.1. Screenshots

The multi-record display



The single-record display


Edit-DB v2.05.015: Database Manager - - - MySQL=localhost ver 5.0.18-nt - - - - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.edit-db.com/EditDBDemo/Table.html
Google
database : demo
object type : Tables
SQL
Re-login
style classic
© Nigel Maddocks, 2000-2007 news help
right-click links for options
Recent hide
employee
Table
activity
employee
employee_history
operation
role
Back
Refresh
Update
Delete
Add
employee
Field name
Field type
Value
ID
INTEGER 11
2
Forename
VARCHAR 50
Benita
Surname
VARCHAR 50
Blood
Role
INTEGER 11
2
Salary
DECIMAL 9,2
12000.00
Activity
INTEGER 11
3
LastChanged
DATETIME 19
2006-01-25 00:00:00.0
Done

2. Walkthrough

This section will take you from logging in, through viewing a table, to editing a record.

The instructions are given assuming you are using the demonstration server on the *demo* database which is accessible from the web-site http://www.edit-db.com

If you are using another server, then you will need to amend the login settings accordingly.

2.1. Logging In

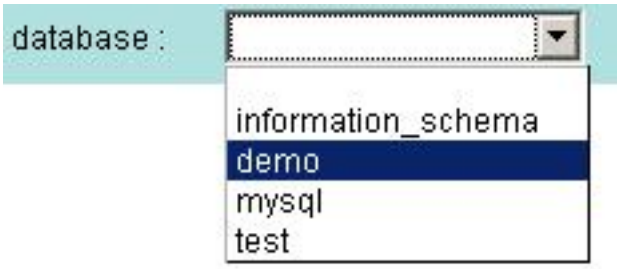
n.b. You will need to enable pop-ups within your browser from the server that is running Edit-DB as logging-in to the database is achieved through a pop-up.

On opening the link to the demonstration system, a blank base page is displayed which immediately throws open the Login dialog box.

Login using the following settings: (password is *demo*)

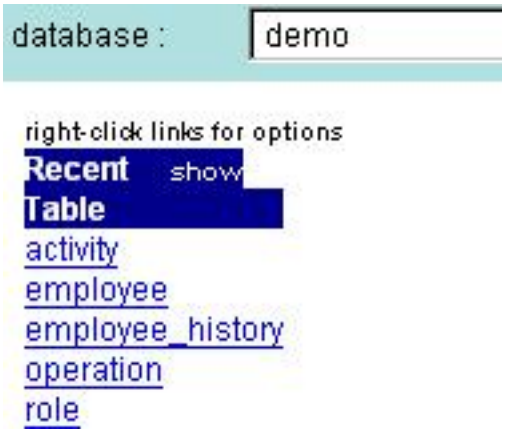
If the login attempt fails, the reasons are displayed in the base page and the Login dialog box is redisplayed

On successful login, the Login dialog disappears and the base page shows two drop-down controls, one for *Database* and the other for *Object type*, which is defaulted to *Tables*. Select the database *demo*.

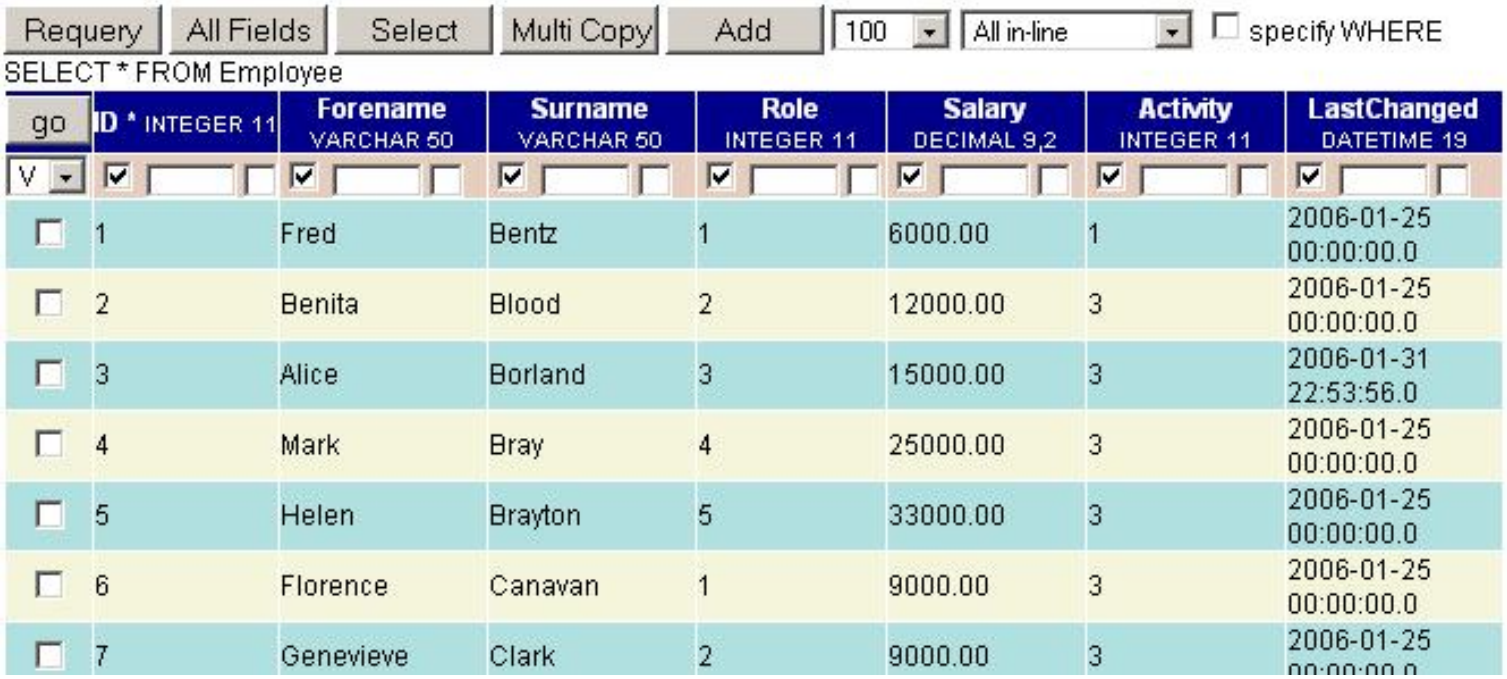


2.2. Viewing a Table

After successfully logging-in, and selecting the database *demo* within the *Object-type* frame, a list of Tables is displayed within the *Object-List* frame.



Now display the *Employee* table by clicking on the *Employee* link. The *Employee* table is shown as follows in the *Object-Detail* frame:

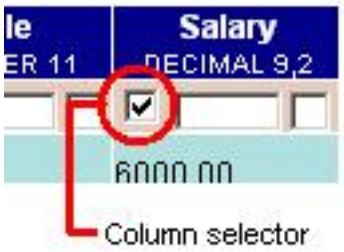


Once a table has been viewed, it will appear in the *Recent* tables sub-list which is toggled to be displayed by clicking on the *show* link.

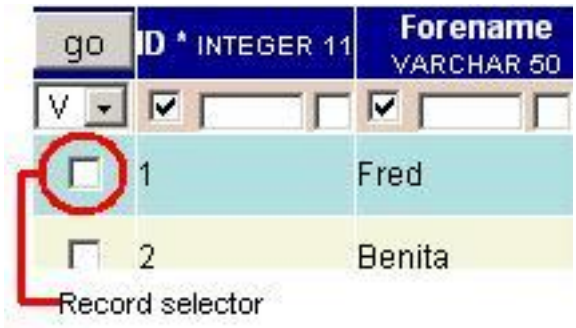
2.3. Viewing a Record

The *Employee* table does not contain many columns, so every column will normally be visible in this *multi-record* display mode. However, many tables in production systems will contain too many columns for them all to be visible in this mode without having to scroll horizontally.

An alternative is to reduce the number of columns being displayed. A quick way of doing this is to un-check the column selector for the columns that you do not want to see. Then press the Requery button.



An easier alternative for viewing individual records is to select a record and view it in the *single-record* display mode. To do this, select the record using the check-box alongside the record, then press the go button.



Alternatively, just double-click the record selector check-box.

The selected record is now displayed in the *single-record* display, still within the *Object-Detail* frame.

It is likely that most records will now be able to be displayed without any scrolling. For records with a large number of columns, it may be necessary to scroll vertically.

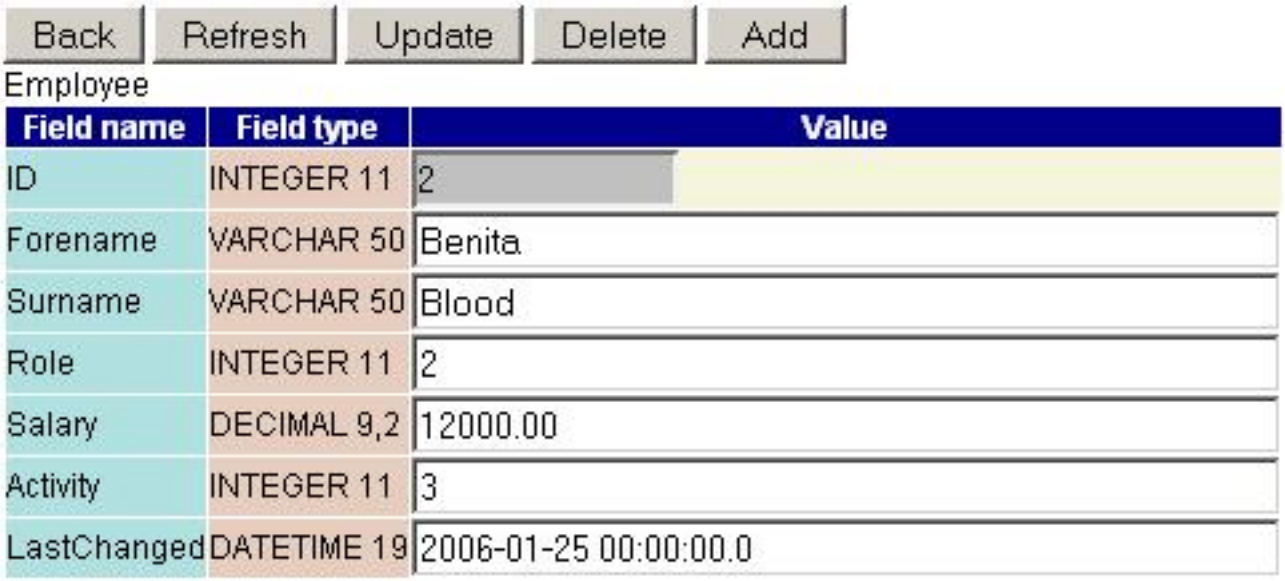
To return to the *multi-record* display, press the 

button within the *Object-Detail* frame. N.B. Pressing the browser's *Back* button gives unpredictable results.

2.4. Editing a Record

Records that can be displayed in the *single-record* display can also be edited by pressing the 

button which re-displays the selected record for editing as follows.



The *Salary* value can be changed to, say, 12500, and the record updated by either pressing *Enter* or the *Update* button.

Other operations that can be performed on this screen include:

- *Record deletion* Using the *Delete* button (which needs confirmation)
- *Record insertion* Using the *Add* button (which also needs confirmation). This is a useful way of creating a new record based on an existing one. Edit-DB recognises some auto-generating columns, like *auto-increment* and presents them with a grey background to indicate that values should not be provided for these columns.
- *Record refresh* Using the *Refresh* button which re-queries the database for the current data for the record.

After updating the record, press the 

button to return to the *multi-record* display.

n.b. The *multi-record* display is not automatically updated with the new values as the *Back* button uses the browser history cache to present the screen. The behavior is by design in order not to put unnecessary demand on the database server. To see the latest data in the *multi-record* display, simply press the *Requery* button.

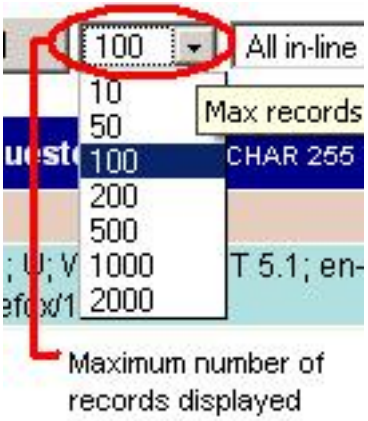
To set the *single-record* display to default to *Edit* mode, then from the *multi-record* display set the *Edit/View* choice to *E*.

3. Tables

* Some of this section also applies to Views

To display a list of tables for a particular database or schema (and owner depending on database type), select *Tables* from *Object type* in the *Object-type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on database type), are completed.

Select a table from the list by clicking on the table name. Up to the first 100 records from the table are displayed by default. A limit is always applied to the number of records displayed but it can be modified by amending the selection at the top of the *multi-record* display.



A database column might be used to contain text, perhaps used for a letter and might contain line breaks. These line breaks will not be rendered as such unless the *select textual display* drop-down is set to *Show new lines*. A database column might be used to contain html in which case it will be rendered as if it were just ordinary html contained within the page - i.e. the html tags will be invisible within the browser. To display the html tags, set the *select textual display* drop-down to *Render html*.

One of the difficulties of viewing long html tables within a web-browser is that the table headings drop off the page as one scrolls down. Edit-DB is different in this respect, as it dynamically keeps the column headings fixed to the top of the screen no matter how far you have scrolled down. In addition, if the table records are printed, the column headings are printed at the top of each page.

When displaying data in the *multi record display*, schema information is often cached in order to improve performance. If you are not seeing recent schema changes, then press the All Fields button to refresh the schema cache.

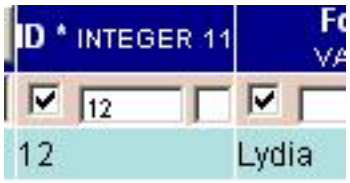
The following sections explain how to define the record selection criteria as well as how to sequence the records. When you move back to displaying this table after viewing another table, the record selection and sequencing settings you had defined for this table are retained and re-applied by default.

3.1. Record selection

* Also applies to Views

Records are selected (or *filtered*) according to record selection criteria which are applied to the columns. Edit-DB enables the user to select records without needing to enter SQL clauses. Underneath each column's heading, there is a small text box into which the user can enter record selection criteria.

For example, to select the record with ID=12, enter *12* into the *record selection criteria* text-box underneath the column-heading for *ID* and press *Enter* (or the *Requery* button).



Whatever is entered into these column-level record selection criteria text-boxes is interpreted into valid SQL and displayed above the output table. In the above example, the generated SQL is

SELECT * FROM Employee WHERE ID = 12

The column-level record selection criteria are AND-ed together, so to display records where Role=2 AND Activity=5, into the record selection criteria text-boxes enter *2* for Role, and *5* for Activity, and press *Enter*. The generated SQL is

SELECT * FROM Employee WHERE Role = 2 AND Activity = 5

Intended selection	**Text-box entry**	**Generated SQL WHERE clause**
IDs 12, 5 and 7	12 5 7	ID in (12,5,7)
IDs between 7 and 10	7-10	ID between 7 AND 10
Forenames starting with F	F%	Forename like 'F%'

Quoting character (CHAR, VARCHAR, TEXT etc) values is usually unnecessary as Edit-DB will apply quotes where necessary. The exception is where the value contains a special character(s), in which case the value should be entered within single-quotes

Special characters are -, %, _, [,], =, >, <, !, like, in, is, not, between, *space*

To select records where a column isn't null, enter *is not null* into the column-level record selection text-box. Similarly to select records where a column is null, enter *is null*.

Although the record selection text-boxes within the *multi-record* display provide easy-access, there will be occasions where they are too small to easily enter all the selection criteria you require, or it may be difficult to see which columns have record selection criteria applied to them. In these cases, it may be easier to enter or visualise the selection criteria on a dedicated web page which is available on pressing the Select button.

Back Search Employee

Field name	Field type	Display	Value	Order
		all none		
ID	INTEGER 11	☑		
Forename	VARCHAR 50	☑	F%	
Surname	VARCHAR 50	☑		
Role	INTEGER 11	☑		
Salary	DECIMAL 9,2	☑		
Activity	INTEGER 11	☑		
LastChanged	DATETIME 19	☑		

This screen actually provides access to the same functionality as the row of check-boxes and miniature text-boxes underneath the column-headings in the *multi-record* display.

Occasionally, the column-level interface for record selection is just not flexible enough to satisfy your requirements. If this is the case then checking the ☑ spec WHERE check-box gives you access to a text-area for specifying the WHERE clause by hand.

e.g.

SELECT * FROM Employee WHERE

```
Forename like 'F%' OR Activity = 2
```

3.2. Record ordering

* Also applies to Views

The easiest way to sort the selected records is to click on the column name in the header section of the displayed table or view. This will initially sort that column into ascending order. Clicking again on the column name will sort the column into descending order. Clicking for a third time on the column name reverts back to ordering into ascending order. This will suffice for the majority of times that ordering is required.

To fine tune the ordering of records, it may be necessary to use the *Order* miniature text-box available for each column next to the column's record selection text-box.

The rules are that a *1* against a column indicates that the column is to be used as the primary "order by" column. A *2* against a column indicates that the column is to be used as the secondary "order by" column, etc. A letter *d* after the number indicates that the column should be sorted in descending order. If you choose to enter the criteria by hand, then press *Enter* or *Requery* to effect the new settings.

The *order by* text-boxes are also available on the *Record Selection Display* shown in the previous section.

3.3. Record editing

* Can also applies to Views (depending on database type)

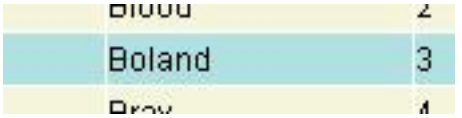
For editing a record using the *single-record* display, see 2.3 Viewing a record and 2.4 Editing a record in the *Walkthrough* section.

3.3.1. In-line editing

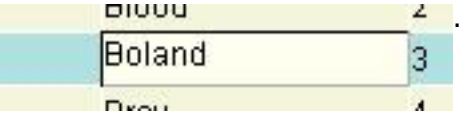
Edit-DB supports in-line editing within the *multi-record* display. To enable this facility set the *Edit/View* default to *Edit*

and the *double-click cell action* to *Edit*

Double-click the record cell that you want to edit, and the display changes as follows from

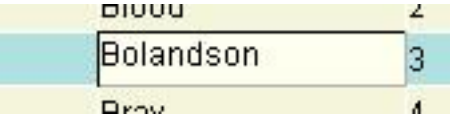


to

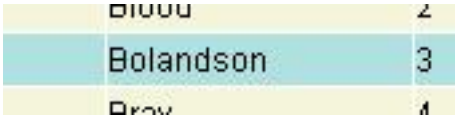


.

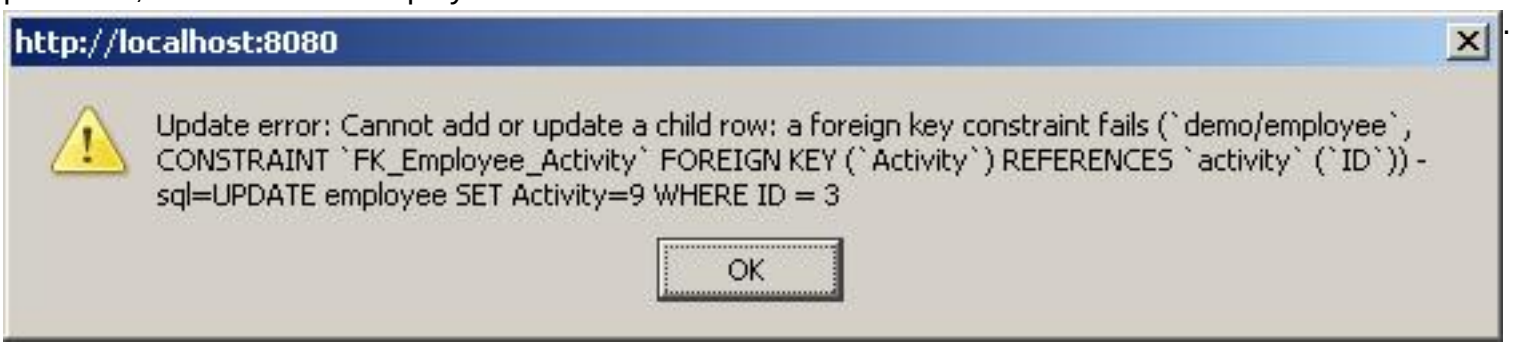
The column value can be changed as desired.



On pressing *Enter*, the record is updated and re-displayed in the usual fashion with its new value.



If there are any errors on update e.g. column not large enough for content, or referential integrity problems, then these are displayed to the user.



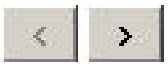
To cancel out of editing a column value, press the *Esc* key.

3.3.2. Navigation within a set of records

Edit-DB has the facility to navigate within a selection of records in order to view and/or edit them.

First, select the records in the set by checking the record-selector check-boxes. Next, ensure the *Edit/View* selector is set to the mode you want as default, then press the *Go* button.

Each record in the set can be viewed and/or edited using the *single-record* display, with movement through the set enabled using the navigation buttons at the top of the screen.



While working with the set of records, each record can be viewed, edited or deleted, or new records added (which will not themselves be included in this set).

After you've finished working with the set or records, press the 

button to return to the *multi-record* display.

3.4. Foreign key look-ups

In-line Display of Summary Data from Record Referenced by Foreign Key

When a table contains foreign keys, the database might also contain Views that pull together the base table columns plus information from the foreign key table. Where this is the case, you might choose to use to display the relevant View (see the Views section).

However there might not be existing Views, and you might not be authorised to create them, or you might prefer to work with the base table at the same time as being able to see information from the foreign key table.

Edit-DB is able to display data from the foreign key table simply by hovering over the foreign key column. The demonstration system contains a table called *Employee* which contains two foreign key columns, *Role* and *Activity* which refer to the primary keys of other tables. By hovering over the the *Role* column within a record, the *Description* from the foreign key table (also called *Role*) is displayed in a tool-tip pop-up.

Just to emphasize, there is no setup required to enable this functionality, simply that there are foreign key relationships defined between tables.

Edit-DB will initially look for a column in the foreign key table called *Description*, ignoring case. Failing that, data is displayed from up to the first three columns containing character data, whose defined length is > 5.

Navigation to Record Referenced by Foreign Key

Alternatively, the record referenced by the foreign key can be viewed simply by double-clicking on the foreign key value, after ensuring that the double-click action is correctly set to *FK rec here*.

The *multi-record* display will be re-positioned onto the referenced table, selecting the referenced record.

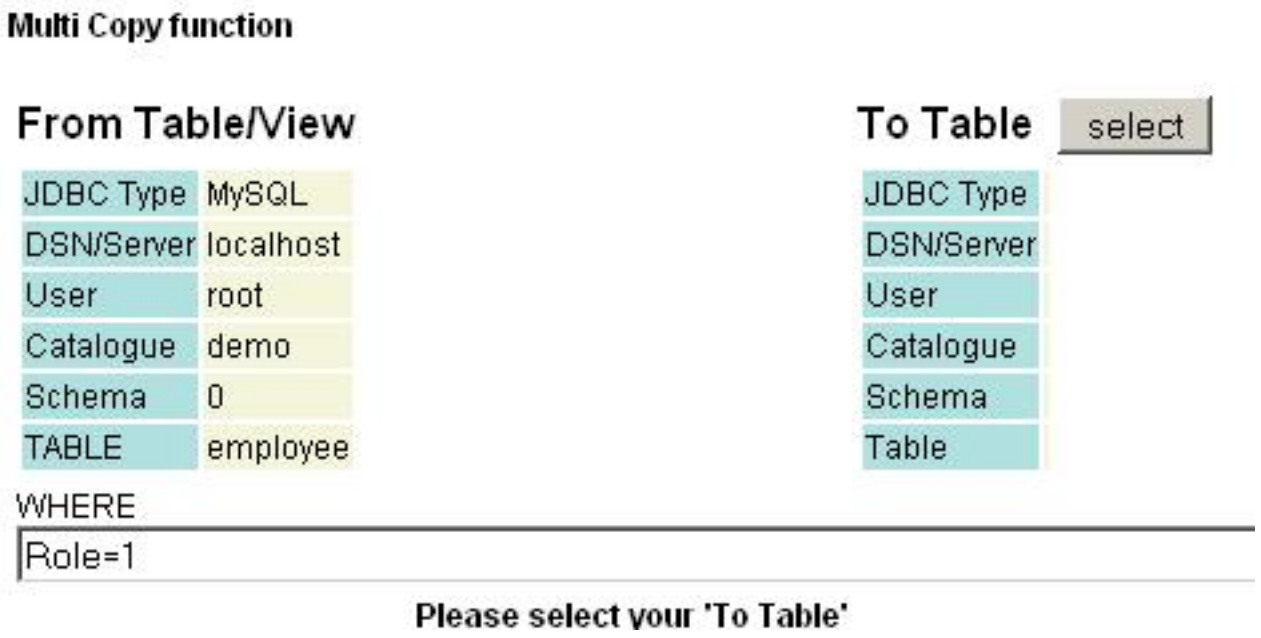
3.5. Multiple-record Copying

* Also applies to copying from Views

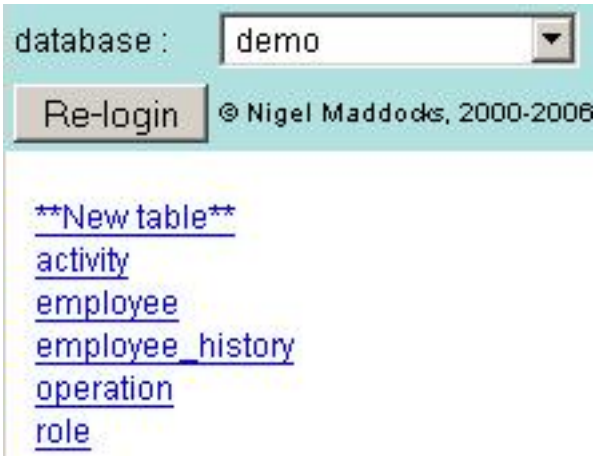
* Please use this functionality with caution - particularly when copying to the SAME table.

Multiple-record copying is accessed through an interface activated after pressing the Multi Copy button.

You are presented with the *From Table* definition and the *To Table* definition which will initially be blank.



Use the *Select* button next to the *To Table*, to select the table to copy to from a pop-up window. You may want to select a table from the same database, or different database, and/or a different database type, and/or a different database server - for which you may need to use the *Re-login* button to select the *To-Database*.

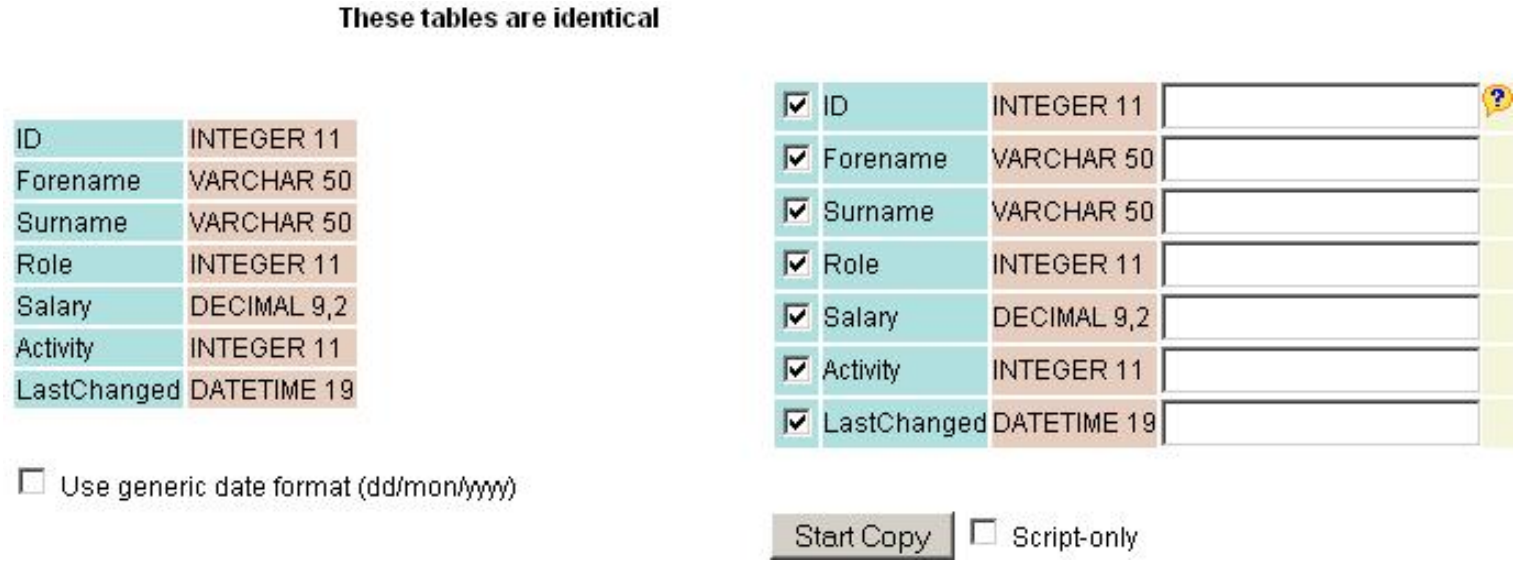


If you want to create a new table to copy into, see Creating a New Table.

Having selected the *To Table*, (in this example the Employee table has been used as a basis for a new table called Employee_copy) you can now define which tables should be copied into, e.g. it might be better not to copy into auto-increment columns, and you can define what data to copy into the *To Table* columns.

Enter either :

- Nothing, to copy the contents of the same sequence column in the *From-table*
- *&column-name*, to copy the contents of the column having this name in the *From-table*
- a value, to set the value of the column to this value for all copied records
- *BLANK, to set the value of the column to blank (rather than null) for all copied records



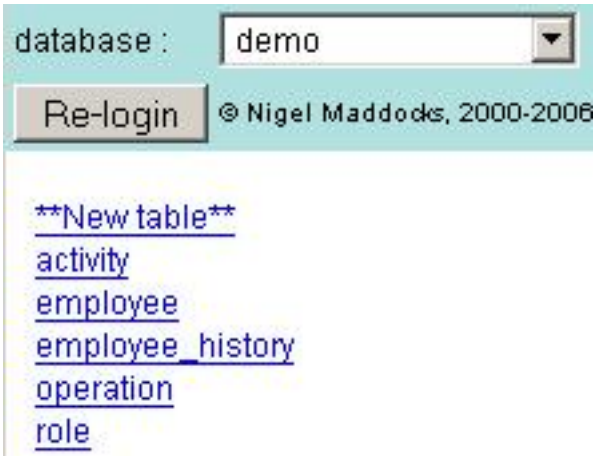
If you are new to this functionality or you want to see the generated SQL statements before having them applied to the database, check the *Script-only* check-box. If this is checked then pressing the *Start Copy* button will provide a link from where the SQL statements are displayed. To return to the original table, click on the table name in the *Tables* list or the *Recent* tables sub-list.

If the *Script-only* check-box is not checked, then on pressing the *Start Copy* button, SQL statments are generated to perform the copy, and then applied to the database, and committed. Copies into Oracle tables are slightly different in that all the SQL statements are applied as a transactional block which is completely rolled-back if any statements fail.

The generated SQL is portable, and can be stored to a file and used to re-create table data at a later date or on another server.

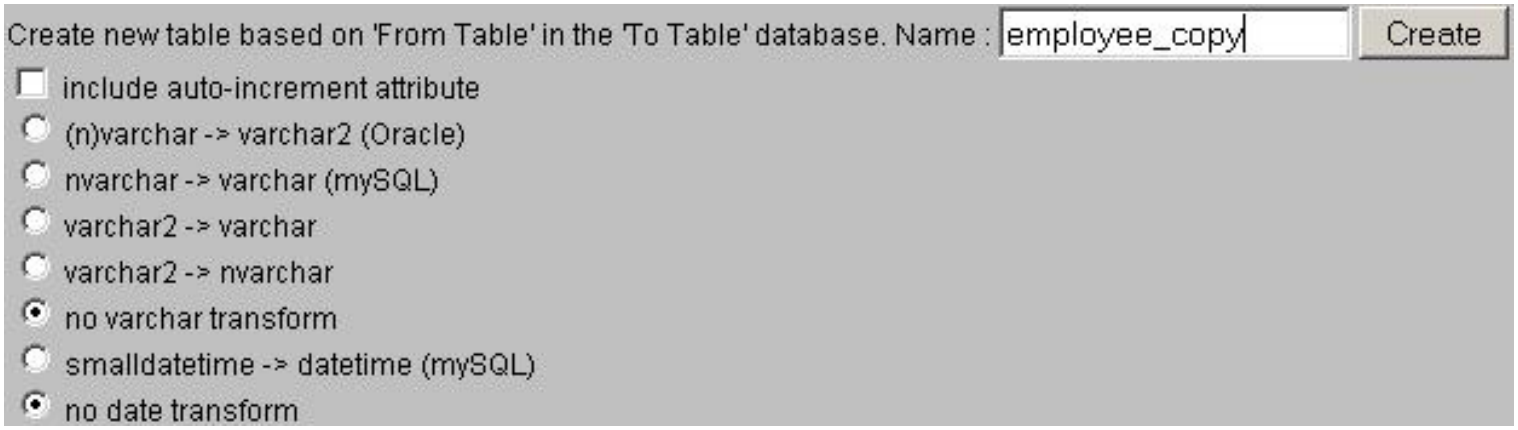
3.5.1. Creating a New Table

If you want to create a new table to copy into then from the *Select To-Table* popup window



select ***New table***.

Enter a name for your new table, then press *Create*.



This is not the best tool for creating tables, so if there are database-specific ways of creating new tables then those might be preferable.

There are some options available for converting between equivalent, but differently named data-types, when creating the new table in a database of differing type, e.g. copying from a MS SQL Server table to an Oracle table.

After having created your new table, return to the *Select To-Table* popup window and select the new table.

3.6. Dates

This section considers Date, Date-Time and Timestamp data types which are all referred to as *date* within this section.

Date data values may be output in a number of formats depending on

- the locale of the server hosting Edit-DB
- the database type
- the locale of the database server
- locale settings within the database
- date formatting options set within the database

Edit-DB does not attempt to influence the format of the displayed date data in order to keep the output similar to that produced by individual database-specific utilities such as MS SQL Server's Query Analyser or Oracle's SQLplus.

When inputting or editing existing data, it is intended that the user should be able to use the same format as that employed when displaying data. This enables the user to simply type over existing column values. If there are any cases where this is found not to work, please contact us.

If you are experiencing problems with date entry then

dd/mon/yy e.g. 12/mar/06 will work with most databases

ccyy-mm-dd e.g. 2006-03-12 will work with MySQL

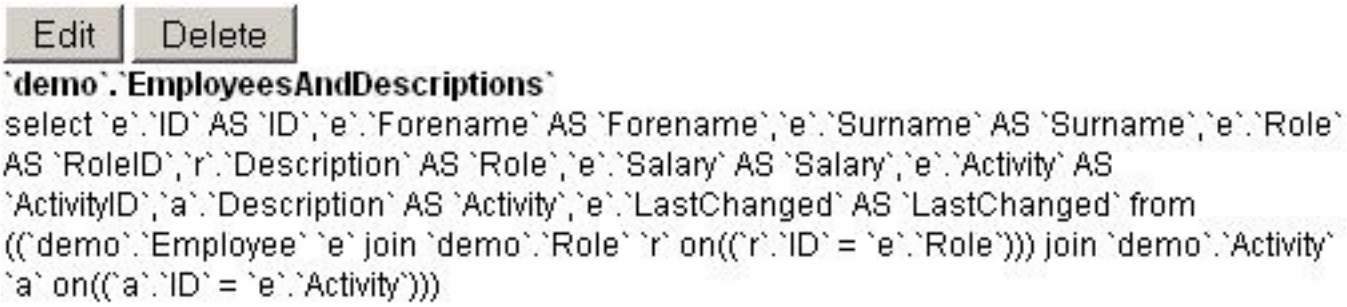
4. Views

To display a list of Views for a particular database or schema (and owner depending on database type), select *Views* from *Object type* in the *Object-type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on database type), are completed.

Most of the functionality applicable to Tables also applies to Views. See the Tables section. Views may also have their records edited if a unique record identifier can be determined.

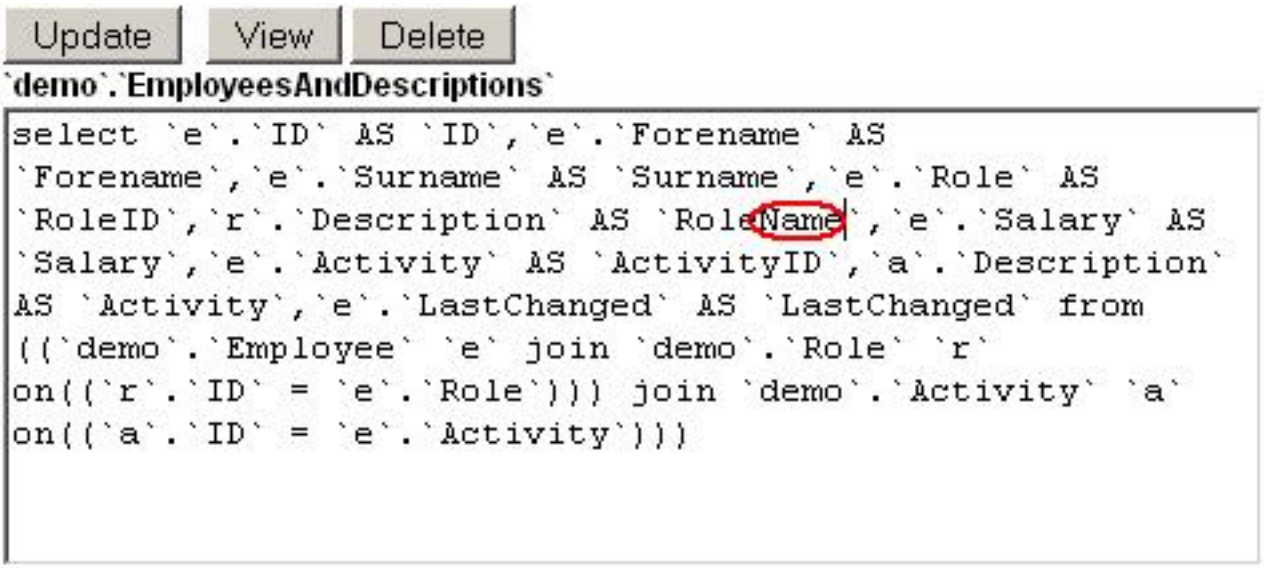
In addition to working with the contents of a View, its definition can be displayed too by clicking on the ***S*** (for Source) link next to the View name.

Oracle and MS SQL Server allow the View definition to be retrieved in the exact format in which it was originally defined. The example here is taken from MySQL, which re-writes the view definition before it is internally stored. The database developer originally did not write the apostrophe delimiters or the excessive apostrophes but did have line breaks to enhance readability.



The View can be deleted by pressing the *Delete* button and confirming.

The View can be edited by pressing the *Edit* button. In this example, the name of a column is changed.



On pressing *Update* the View definition is updated. The output records from the View can be displayed in the usual way by clicking on the View name in the *Object-List* frame. If the View update fails, then the error message from the database is displayed, and the view definition can be further modified.

Note: Oracle will often accept invalid object definitions and only present the problems when the object is used.

When displaying data in the *multi record display*, schema information is often cached in order to improve performance. If you are not seeing schema changes (as in this case, changing an output column name within a view), then press the 

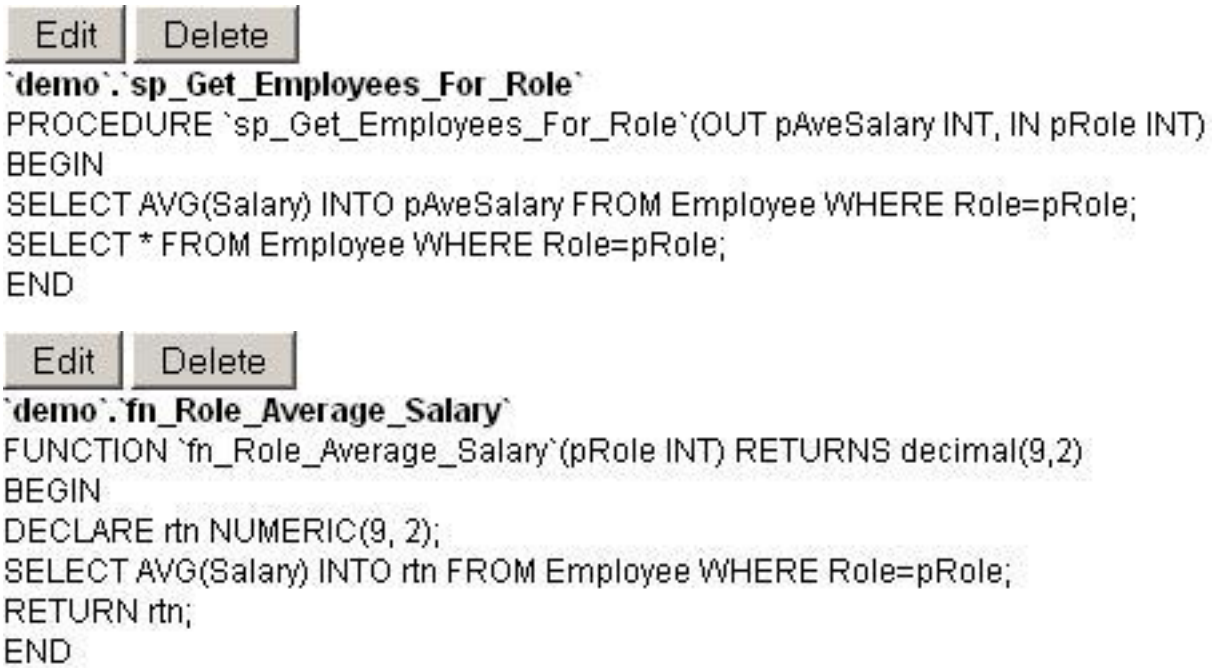
button to refresh the schema cache.

5. Procedures and Functions

To display a list of Procedures or Functions for a particular database or schema (and owner depending on database type), select *Procedures* or *Functions* from *Object type* in the *Object-type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on database type), are completed.

The Procedure or Function definition can be displayed by clicking the listed object in the *Object-List* frame.

n.b. Postgre defines a Procedure as a Function with no return value, and can have multiple functions with the same name differentiated by the parameter list.



The Procedure or Function can be deleted by pressing the *Delete* button and confirming.

The Procedure or Function can be edited by pressing the *Edit* button.

Update View Delete

`demo`.`sp_Get_Employees_For_Role`

```
PROCEDURE `sp_Get_Employees_For_Role`(OUT pAveSalary INT,
IN pRole INT)
BEGIN
SELECT AVG(Salary) INTO pAveSalary FROM Employee WHERE
Role=pRole;
SELECT * FROM Employee WHERE Role=pRole;
END
```

Update View Delete

`demo`.`fn_Role_Average_Salary`

```
FUNCTION `fn_Role_Average_Salary`(pRole INT) RETURNS
decimal(9,2)
BEGIN
DECLARE rtn NUMERIC(9, 2);
SELECT AVG(Salary) INTO rtn FROM Employee WHERE
Role=pRole;
RETURN rtn;
END
```

On pressing *Update* the Procedure or Function definition is updated. If the update fails, then the error message from the database is displayed, and the definition can be further modified.

Note: Oracle will often accept invalid object definitions and only present the problems when the object is used.

6. Triggers and Indexes

To display a list of Triggers or Indexes for a particular database or schema (and owner depending on database type), select *Triggers* or *Indexes* from *Object type* in the *Object-Type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on database type), are completed.

A Trigger or an Index is related to a particular Table. The list shows all Triggers or Indexes within a sub-heading of the Table to which they relate.

The list of Indexes is further enhanced by displaying, to the right of the index name, the columns comprising the index. The frame border between the *Object-List* frame and the *Object-Detail* frame may need to be moved to the right in order to see this information. Columns know to form part of the primary key are suffixed *(PK)* and those comprising part of a unique index are suffixed *(U)*.

Edit-DB

Edit-DB ver 2.05.016, Doc ver 2.05.016

---

right-click links for options

Index

activity

ID	ID

employee

FK_Employee_Activity	Activity
FK_Employee_Role	Role
ID	ID

employee_history

FK_Employee_History_Activity	Activity
FK_Employee_History_Operation	Operation
FK_Employee_History_Role	Role
ID	ID

operation

ID	ID

role

ID	ID

The Trigger or Index definition can be displayed by clicking the listed object in the *Object-List* frame.

Edit | Delete

`demo`.`tr_Employee_Update`

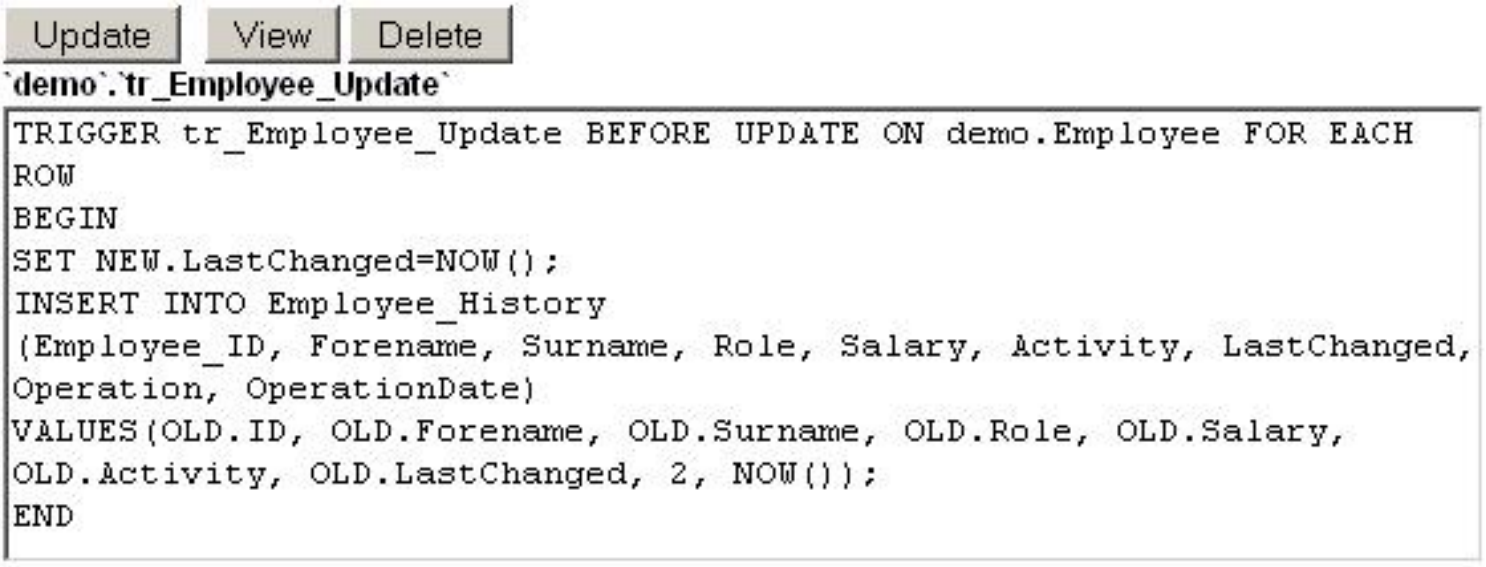
```
TRIGGER tr_Employee_Update BEFORE UPDATE ON demo.Employee FOR EACH ROW
BEGIN
SET NEW.LastChanged=NOW();
INSERT INTO Employee_History
(Employee_ID, Forename, Surname, Role, Salary, Activity, LastChanged, Operation, OperationDate)
VALUES(OLD.ID, OLD.Forename, OLD.Surname, OLD.Role, OLD.Salary, OLD.Activity, OLD.LastChanged, 2,
NOW());
END
```

Properties of index FK_Employee_Activity on Employee

```
CREATE INDEX FK_Employee_Activity USING BTREE ON demo.Employee(
Activity)
```

The Trigger can be deleted by pressing the *Delete* button and confirming.

The Trigger can be edited by pressing the *Edit* button.



On pressing *Update* the Trigger definition is updated. If the update fails, then the error message from the database is displayed, and the definition can be further modified.

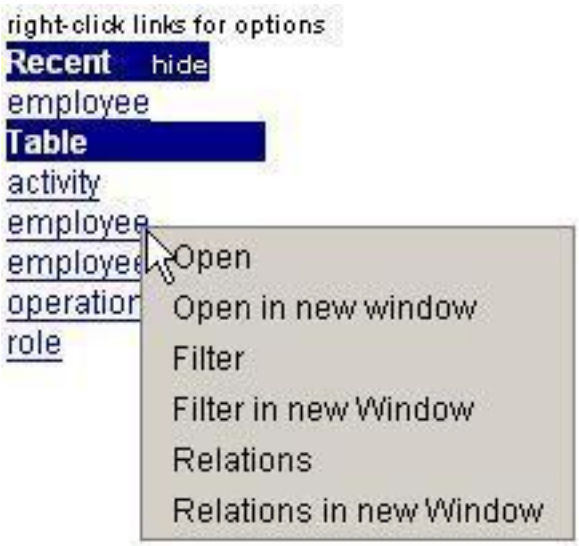
Note: Oracle will often accept invalid object definitions and only present the problems when the object is used.

Edit functionality is not yet available for indexes.

7. Relationships

To display information on Relationships between tables and columns, first select *Tables* from *Object type* in the *Object-Type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on database type), are completed.

This results in a list of tables in the *Object-List* frame. Bring up the context menu by right-clicking the



Clicking on a *Relations* leads to a listing of that table's foreign key relationships, and a listing of tables where the selected table is being used for another table's foreign key(s).

Each listed table is also clickable so that its relationships can be viewed.

Relationships for **employee**

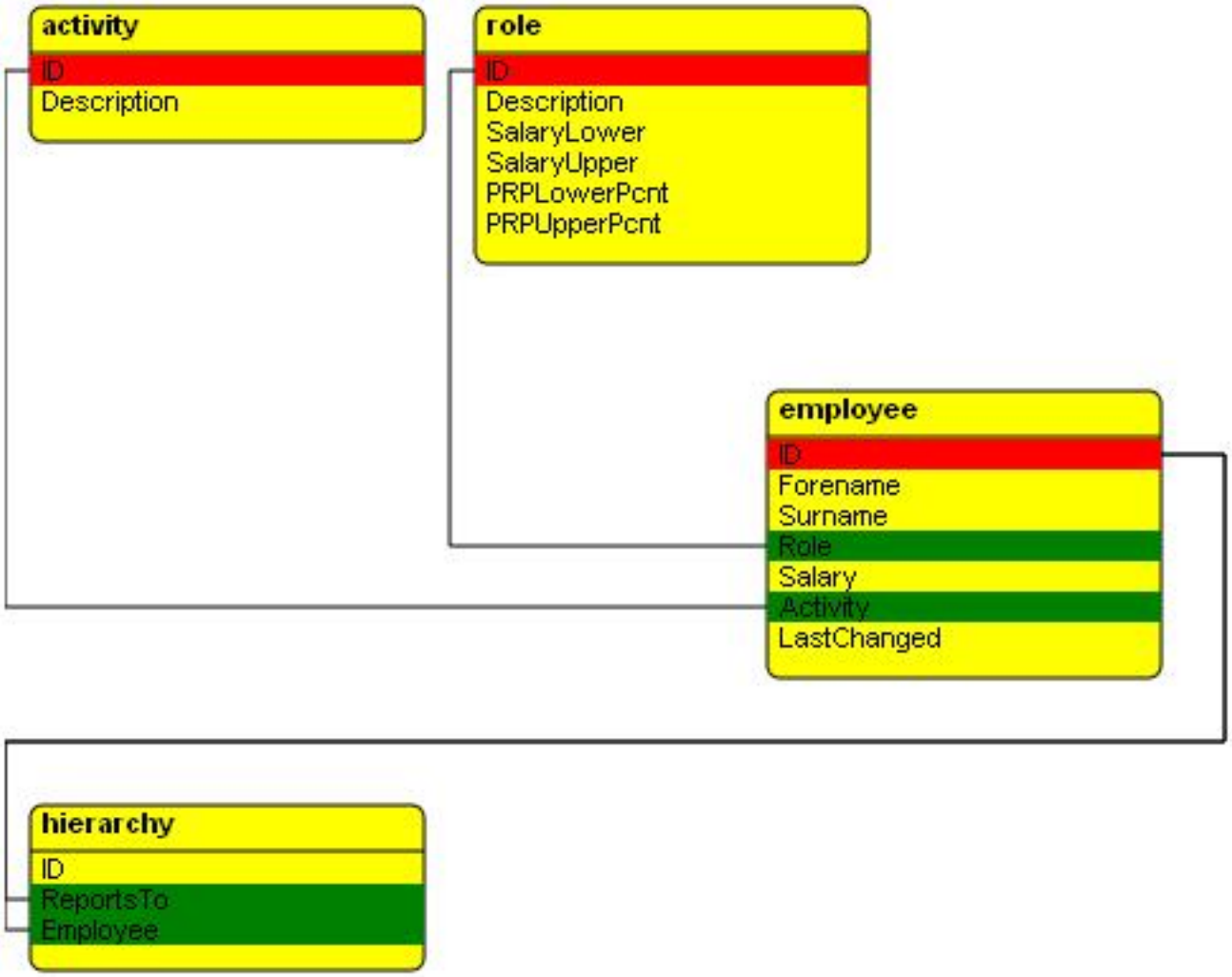
Foreign Keys (columns in this table which must match column values on other tables)

Foreign column	Primary Key Table				Relation name
	Catalog	Schema	Table	Column	
Activity	demo	null	activity	ID	FK_Employee_Activity
Role	demo	null	role	ID	FK_Employee_Role

Primary Key References (this table's primary keys and what other tables' columns are referencing them)

Primary column	Foreign Key Table				Relation name
	Catalog	Schema	Table	Column	
ID	demo	null	hierarchy	Employee	FK_Hierarchy_Employee_Employee
ID	demo	null	hierarchy	ReportsTo	FK_Hierarchy_ReportsTo_Employee

For Firefox versions 1.5 and above and Internet Explorer with the Adobe SVG Viewer plug-in installed, the listed relationships are also displayed graphically using Scalable Vector Graphics (SVG).



- It was hoped that Internet Explorer 7 would support SVG natively, but from reports about the latest beta, this appears to not be the case, which would mean that a third-party plug-in would still be required.
- Firefox supports SVG natively in versions 1.5 and above. Printing of SVG diagrams has some problems particularly for large diagrams. An earlier printing bug https://bugzilla.mozilla.org/show_bug.cgi?id=314707 has now been resolved.

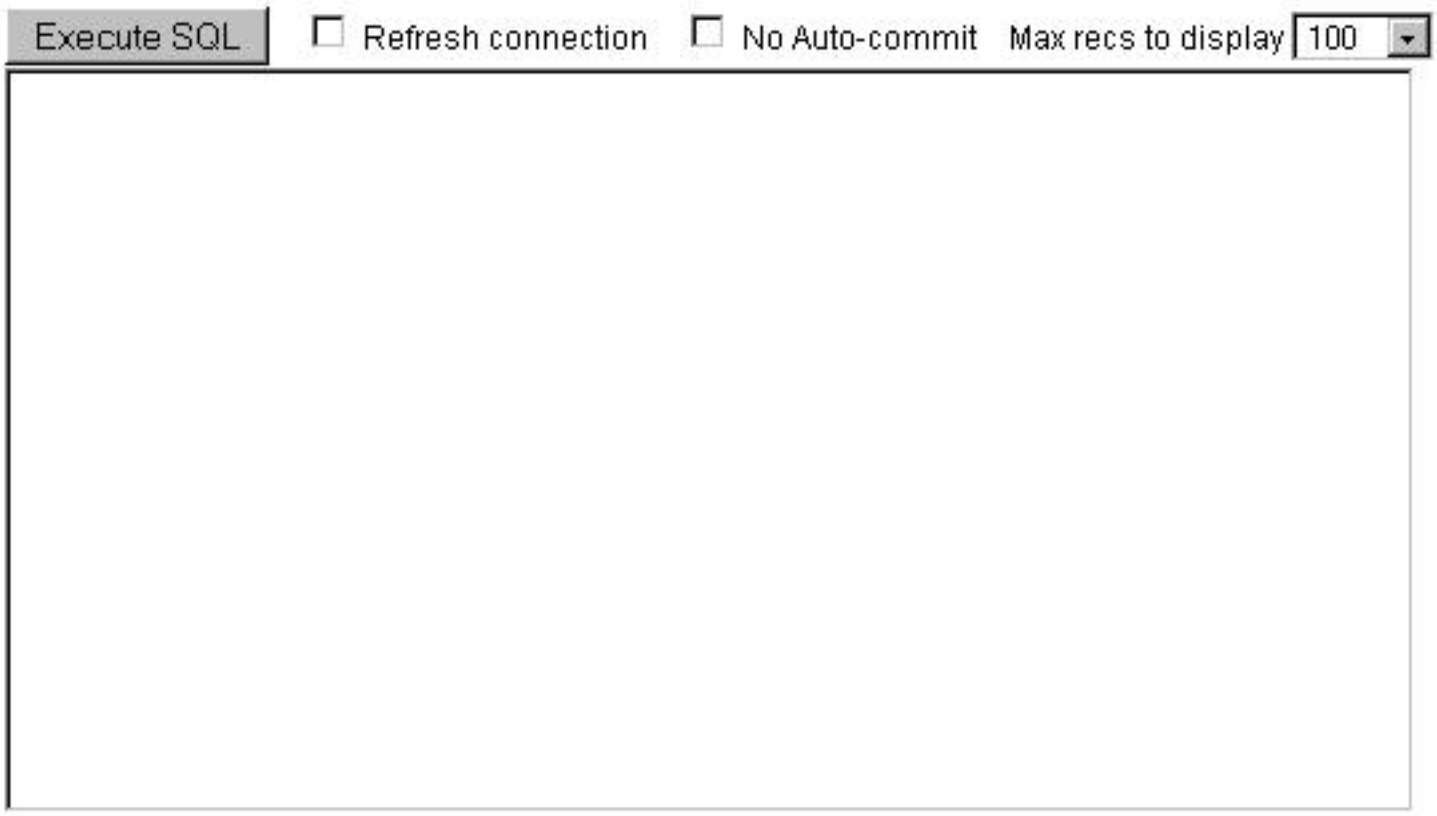
8. SQL Entry

To access the SQL Entry facility, select *SQL Entry* from *Object type* in the *Object-Type* frame, having ensured that the other selections to the left, labelled *Database* or *Schema* (and *Owner* depending on

database type), are completed.

The SQL Entry area will appear in the *Object-Detail* frame. To open it in a new window, use the SQL button. This enables SQL to be entered in one window while keeping the ability to browse database objects in the other.

The Free Text SQL display appears as follows:



The text-area size is proportioned to appear as large as possible while remaining within the browser viewport. If the browser is re-sized, then the text area will be re-sized after the next page submission, i.e. after *Execute SQL* is pressed.

The SQL entered must conform to the allowed statements and syntax of the database type, otherwise an error will be reported back to you.

Common statements might be SELECT, UPDATE, INSERT, DELETE statements, or CREATE TABLE,

ALTER TABLE instructions.

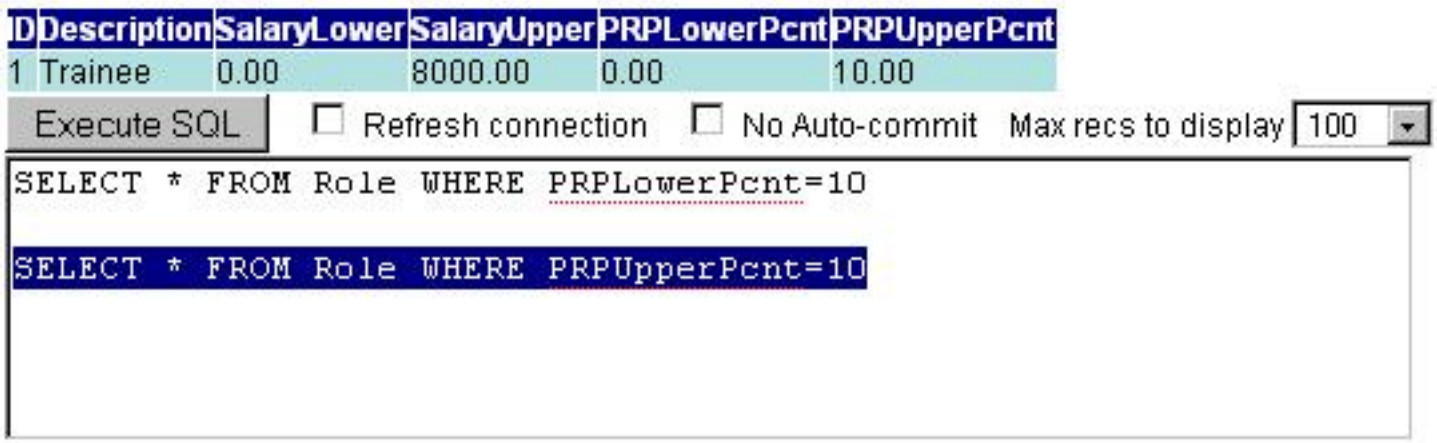
SELECT statments should return records which will be displayed on the page.

UPDATE and DELETE statements should affect a number of records. This number is displayed on the page.

Some database types allow the entering of a number of SQL statements which are run sequentially, by the database server.

In order to run a number of SQL statements in Oracle, enclose the statements within BEGIN ...statements... END; and ensure that each statement ends in a semicolon.

The text-area can be used as a scratch-pad where a number of SQL statements are jotted down. The particular statement to be run can be selected by highlighting it with a mouse or keyboard before pressing *Execute SQL*.



Calling Stored Procedures

Stored Procedures should be called using the syntax

CALL qualifier.stored-procedure-name(parameter1, ..., parameterN)

The qualifier might be unnecessary, or might need to be the schema name, or in the case of Oracle, where the stored procedure might reside within a package, the package name.

The values for *output* parameters and return values can be displayed if, in the correct positions, the

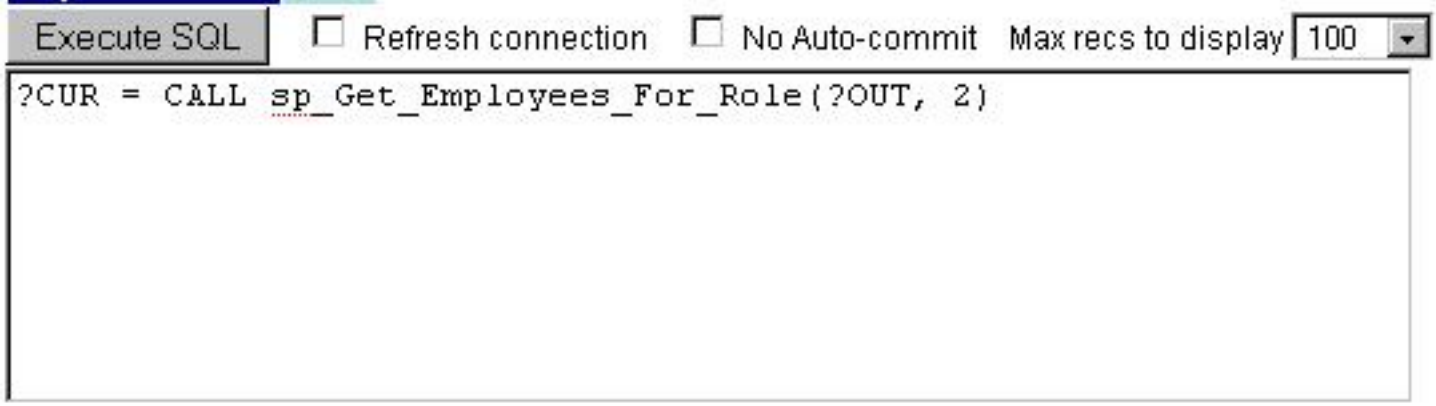
following substitution values are used:

- **?OUT** for a parameter/return value of any scalar type
- **?CUR** for a parameter/return value of a cursor or resultset/recordset type

For example, the procedure sp_Get_Employees_For_Role returns a recordset or employees for the parameter Role, but also has an output (int) parameter containing the average salary for that role. Both of these outputs can be displayed by calling the procedure using this syntax.

ID	Forename	Surname	Role	Salary	Activity	LastChanged
2	Benita	Blood	2	12000.00	3	2006-01-25 00:00:00.0
7	Genevieve	Clark	2	9000.00	3	2006-01-25 00:00:00.0
12	Lydia	Glaser	2	16000.00	5	2006-01-25 00:00:00.0
17	Charles	Irving	2	12000.00	3	2006-01-25 00:00:00.0
22	Irma	Kreutzer	2	12000.00	3	2006-01-25 00:00:00.0
27	Loren	McKinney	2	16000.00	3	2006-01-25 00:00:00.0
32	Zetta	Plantz	2	13000.00	3	2006-01-25 00:00:00.0
37	Beatrice	Ullman	2	13500.00	5	2006-01-25 00:00:00.0

Output Parameter 12938



CALL or *EXEC* can be in upper or lower case. *?CUR* and *?OUT* can be in upper or lower case. *?CUR* can be replaced by *:X0* ... *:X9* in upper or lower case and without initialisation.

The *No Auto-Commit* check box forces the SQL to be run without automatically committing any results.

The *Refresh Connection* check box forces the SQL to be run using a new database connection, rather than the cached connection. This can be useful when using SQL to call Oracle Stored Procedures which are under current development. In normal circumstances, there is no need to use a new database

connection.

9. Other features

Selecting the Style

The application can be rendered in alternative styles by selecting from the style choice drop-down in the *Object-Type* frame

Opening the *Object-Detail* frame in its own Browser Window

The object selected for viewing from the object listing in the *Object-List* frame, is displayed in the *Object-Detail* frame. It can be opened in its own browser window by using the context menu and selecting an *...in new Window* option. This feature is particularly useful when wanting to compare the contents of different tables. A number of tables can be opened in their own browser windows and re-sized and positioned to give a tiled view.

n.b. Style changes will not be applied to child browser windows which are already open.

Other Object Types

The *Object type* selector may contain other options such as *Local Temporaries* which have not been individually documented here. They should be treated as tables in the first instance. If you consider that there are object types of particular interest to you, which should be treated in a particular manner, then please contact us.

Row Highlighting in Tables and Views

When making comparisons between rows in the *multi record* display, one often has to scroll horizontally as the records may be quite wide. Rows can be highlighted to help comparison by double-clicking in the cell which contains the row-selector. Take care not to double-click the check-box itself otherwise the row appears in the *single record* display. The highlighted row will revert to normal if the row-selector cell is double-clicked again.

10. Database Type Specifics

10.1. MySQL

Ports

The default port is 3306. To use another port in a database connection, append it to the *Server* setting, separated by a colon, in the login dialog, i.e.

servername:port or

serverIP:port

Views, Procedures, Functions, Triggers and Indexes are only available for listing and maintenance in MySQL versions 5+ because of the evolution of MySQL.

10.2. Oracle

Ports

The default port is 1521. To use another port in a database connection, prefix it to the *Instance* (or System Identifier - SID) setting, separated by a colon, in the login dialog, i.e.

port:instanceName

Connection Identification

Connections from Edit-DB have the signature *EditDB s=server-name c=client-ip date&time of connection start*. This can be seen in the *PROGRAM* column of the system View *v$session*.

Database Sequences

Database Sequences can be selected as an object type and listed in the ususal way. Their current values and increments are displayed.

10.2.1. Oracle Packages and Package Body

Oracle Stored Procedures and Functions are often not held individually within the database, but instead within a Package. If the procedures and functions are directly used from a client, then their names should be qualified by the Package name (*package.procedure*).

Procedures and Functions residing within a Package have their logic defined in a *Package Body*. Such objects can refer to each other without needing qualification, and any such object can make calls on any other object within the same Package. Any Procedures and Functions that are to be used by external clients must be declared in a *Package*.

Packages are listed when selecting *Packages* in the *Object type* drop-down in the *Object-Type* frame.

Package Bodies are listed when selecting *Package Body* in the *Object type* drop-down in the *Object-Type* frame.

Packages can be viewed, edited and deleted using that standard buttons. They should only contain the signature of the procedures or functions (or other objects) i.e. the object name, parameters and return type.

Package Bodies can be viewed, edited and deleted using that standard buttons. They should contain the full definitions of the objects contained within. When Viewing a Package Body, the whole object is scanned for Procedure and Function definitions. Where they are found, links are displayed at the top of the Package Body to enable quick navigation to the individual Procedures and Functions.

Changes to Packages and Package Bodies can be saved without their contents being valid. Any invalid Packages, Package Bodies, Stored Procedures, Functions, Sequences, Object Types or Collection Types are listed with Invalid! to the right of the name. When viewing the object definitions, any *Invalid* objects have their known errors listed before their definitions. If there are no errors displayed for an invalid object, then a re-compilation might solve the issue. An object can be recompiled by pressing the buttons *Edit* followed by *Update*.

10.2.2. Oracle Types and Collections

Oracle developers find *Object Types* and *Collection Types* (or VARRAYs) to be useful, particularly when passing data to and from Stored Procedures.

Oracle *Types* and *Collections* can be listed by selecting *Types* from the *Object type* drop-down in the *Object-Type* frame. A list of *Object Types*, where the name is appended with (*object*), and *Collection Types*, where the name is appended with (*collection*), is displayed.

On selecting an object, its definition is displayed, and buttons allow it to be amended or deleted.

10.3. SQL Server

Ports

The default port is 1433. To use another port in a database connection, append it to the *Server* setting, separated by a colon, in the login dialog, i.e.

servername:port or

serverIP:port

Connection Identification

Connections from Edit-DB have the signature *EditDB s=server-name c=client-ip date&time of connection start*. This can be seen in the *PROGRAM* or *APPLICATION* column of the *current connections* within MS SQL Server *Enterprise Manager*.

10.4. Postgre

Ports

The default port is 5432. To use another port in a database connection, append it to the *Server* setting, separated by a colon, in the login dialog, i.e.

servername:port or

serverIP:port

Database Sequences

Database Sequences can be selected as an object type and listed in the ususal way. Their current values and increments are displayed along with other information.

Database Name

When using the login dialog, the database name should be specified. If it is not provided then the database name *postgre* will be substituted.

Triggers

Postgre triggers can not currently contain SQL statements. Instead a trigger should call a function. This is a feature of the database.

10.5. IBM AS/400, iSeries

The use of Edit-DB with IBM AS/400 or iSeries databases is not supported as we do not currently have access to an AS/400 database. However the basic operations do work and people may find it useful.

Specifying the Library

The *library* should be manually entered in the *Object-Type* frame. A drop-down list is not provided as the AS/400 takes too long to provide the list.

Multi-Member Files

The AS/400 has the concept of *members*, which are essentially child tables having an identical structure to the parent table. The parent table, in this analogy, holds no data, only defining the structure. If the concept of members is not actively used, then actions on the table will default to acting on the first member. It is unusual for a table to have no members, but if this is the case, it can not hold any data.

In Edit-DB, once the table (file) has been selected, data from the first member is displayed. To change the member being worked with, enter the name of the member in the associated text-box.

10.6. IBM DB2

Ports

The default port is 50000. To use another port in a database connection, append it to the *Server* setting, separated by a colon, in the login dialog, i.e.

servername:port or

serverIP:port

User IDs

User IDs are not simply database entities - they must also exist as server users e.g. a Windows User or Linux User.

Error Messages

Error messages reported by the database are not as user-friendly as those from other databases. Error numbers can be matched against the DB2 reference documentation found here (Reference --> SQL).

10.7. HSQLDB (Hypersonic)

Ports

The default port is 9001. To use another port in a database connection, append it to the *Server* setting, separated by a colon, in the login dialog, i.e.

servername:port or

serverIP:port

Support

Support for HSQLDB (Hypersonic) is not as mature as for other database types. The object types currently supported are Tables and Views.

Connections

Edit-DB can connect to an HSQLDB database server. It cannot open up a filestore itself as a database. It does not use secure or http protocols. A server and database (a.k.a. alias) must be specified in the login

dialog. The database must match one set up in the HSQLDB server in the server.properties file. Edit-DB does NOT create a new database if the specified one does not exist.

11. FAQ

11.1. Record selectors not visible

Question

Why are no record selectors displayed against my records?

Answer

There needs to be something unique about each record for record-level functionality to be enabled. For AS/400 (iSeries) and Oracle, there are built-in unique record identifiers in the RRN and ROWID respectively, therefore the record selectors are always displayed for tables in these databases.

Other databases' tables will have record selectors displayed if at least one of the following is detected for the particular table

- An auto-increment column
- A primary key
- A unique index

11.2. Back and Refresh buttons do not work correctly

Questions

1. Why does the browser Back button not work correctly?
2. Why does the browser Refresh button not work correctly?
3. Why does the application Back button lead to an empty screen?

1. Answer

The brower's *Back* button is not supported in Edit-DB. If it works then fine, if it doesn't then don't use it.

2. Answer

The brower's *Refresh* button is not supported in Edit-DB. Several screens have their own *Refresh* buttons which will refresh the screen with the current situation from the database.

3. Answer

Sometimes, when returning from a *single-record* display screen to a *multi-record* display, the screen appears blank, or requests a refresh. This is usually caused by the browser cache being full. The resolution is to clear the browsers cache.

For Internet Explorer, this is found in Tools -> Internet Options -> General -> Temporary Internet Files -> Delete Files

For Firefox 1.5+, this is found in Tools -> Options -> Privacy -> Cache -> Clear Cache Now

Edit-DB uses the browser cache when returning to the *multi-record* display screen in order to minimise the need to perform costly queries on the database.