SQLeo Beginner Users Guide

**Revised: 14/06/2012 by Alan Shiers**

**Revised: 31/10/2014 by PAscal**

Table of Contents

[Overview 3](#_Toc422773963)

[Starting SQLeo with the choosen Look and Feel 3](#_Toc422773964)

[Defining JDBC Drivers 3](#_Toc422773965)

[Creating database connection (datasource) 4](#_Toc422773966)

[Metadata explorer 7](#_Toc422773967)

[The Metadata browser mode 7](#_Toc422773968)

[The Metadata Search Feature 8](#_Toc422773969)

[Command Editor 9](#_Toc422773970)

[Using command editor result in text mode 9](#_Toc422773971)

[Using command editor result in grid mode 10](#_Toc422773972)

[New command 10](#_Toc422773973)

[The Visual Query Designer 11](#_Toc422773974)

[The Designer Mode 11](#_Toc422773975)

[The Syntax Mode 17](#_Toc422773976)

[The PREVIEW Window and Query Results 18](#_Toc422773977)

[Adding More Tables To A Query 21](#_Toc422773978)

[The Content Window 24](#_Toc422773979)

[Displaying data of a Table (or a query) 24](#_Toc422773980)

[Sorting Data 26](#_Toc422773981)

[Inserting and Deleting Records from a Table 28](#_Toc422773982)

[Filtering Data 32](#_Toc422773983)

[Finding Terms 34](#_Toc422773984)

[The DEFINITION Window 34](#_Toc422773985)

[SQL History 35](#_Toc422773986)

[Data comparer 36](#_Toc422773987)

[Schema comparer 36](#_Toc422773988)

[Troubleshooting 36](#_Toc422773989)

[Support 36](#_Toc422773990)

# Overview

SQLeo serves as a utility to allow connection to multiple RDBMS (rational database management systems). While SQLeo has powerful features we know you are eager to start using, assumptions are being made before you begin using it. The following topics serve as prerequisite understanding prior to using SQLeo. This guide will cover most of the basic features targeting the beginner user and while SQLeo has many advanced features, these may be covered in another guide for advanced power users.

## Starting SQLeo with the choosen Look and Feel

Command line ...

Look and feel, cf lignes de commande

Redémarrage requis pour en changer

Préférences

* Link vers le detail
* Language
* Font size
* Icon size
* ...

Portable app

Support for Unicode exports / imports

## Defining JDBC Drivers

For all other RDBMS, In order for SQLeo to connect to any given RDBMS, you need to provide a set of JDBC (***J***ava ***D***ata***b***ase ***C***onnectivity) Drivers which serve the function as a communications bridge between SQLeo and the database system. Depending on the RDBMS you are trying to access, you can usually obtain JDBC Drivers from the vendor’s website. The JDBC Drivers are really a set of Java classes that are bundled inside a file with the *jar* file extension. As an example, if you wanted to obtain the JDBC Drivers to a MySQL database, at the time of this writing, you would navigate your browser to this URL: <http://dev.mysql.com/downloads/connector/j/> and download the file mysql-connector-java-5.1.20.zip. Using a ZIP Archive utility, you would then extract the contents of the zipped archive file which would contain a file named: mysql-connector-java-5.1.20.jar. You would place the file mysql-connector-java-5.1.20.jar in a directory where you collect JDBC Drivers for all the databases you intend to connect to through SQLeo.

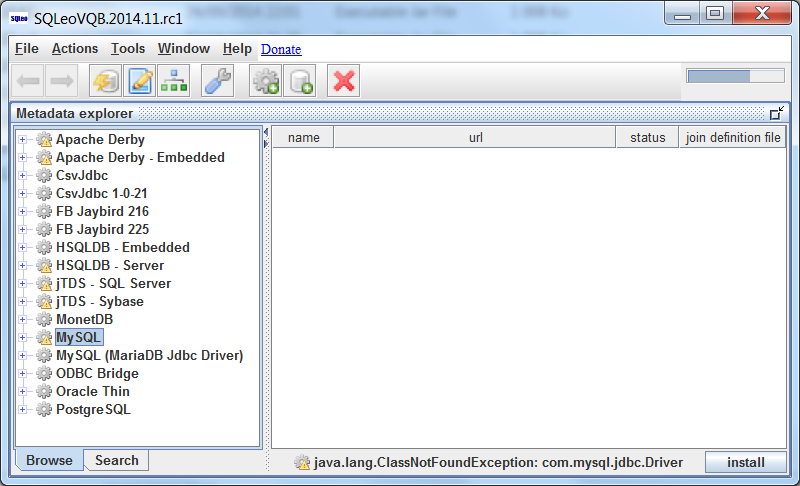
When you first launch SQLeo, you are presented with an interface that displays a list of supported database systems on the left hand pane of the Metadata Explorer(many other RDBMS also supports JDBC. They can be added using the menu *New driver* icon or right click in the left hand pane). See IMAGE 1. If you select any one of the items in the list, a message will appear at the bottom of the interface stating that it could not find the JDBC Drivers for that particular database system. The message refers to a *ClassNotFoundException* and names the file it requires*.* Next to the message is a button labelled “install” that you can use to launch a dialog box that allows you to navigate to the directory on your hard drive where you are storing your JDBC Drivers.

Note: Mariadb jdbc driver used to connect for both MariaDB AND MySQL is already bundled in SQLeo.jar

* Offers multischema view of thoses databases

CsvJdbc driver is provided in lib directory and can be used to read csv file as tables (no need to go on internet to download it, except if you need a newer version ...)

IMAGE 1

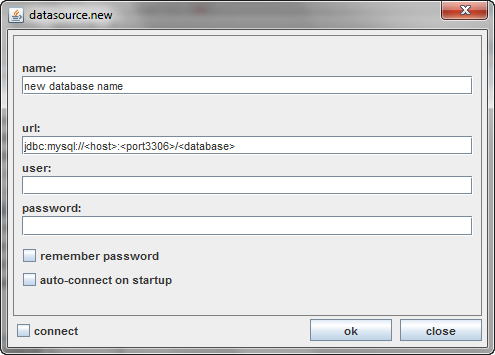


## Creating database connection (datasource)

Once you have told SQLeo where to find the Drivers for the database system you are attempting to connect to, you can then provide SQLeo with further information it requires to make a connection. To do this, you need to launch a dialog box by clicking on the button with the image: C:\SQLeonardo\SQLeo\src\images\database_add.png and labelled *new datasource.* In the case of connecting to a MySQL database system, you will be presented with the following dialog window:

À revoir avec mariaDB

IMAGE 2



In the field labelled *name,* type a new name for the database you are connecting to. In the field labelled *url* edit the existing string: jdbc:mysql://<host>:<port3306>/<database>

This URL string requires you to replace those parts that are in brackets: <…>

The part labelled <host> is where you place the domain name where the database system resides on the network or over the internet. Typically this would follow the pattern such as: www.someplace.com or it could be an IP Address. If the database resides on your computer and not on the network, then you would replace <host> with the term: localhost or 127.0.0.1

The part labelled <port3306> is the port on which the database listens to incoming requests. Though this can be changed by an administrator, the port number by default is 3306. The port number will be different depending on the RDBMS.

The part labelled <database> would be the actual name given to the database. If it was named *mydb* then that is what you would enter (This part has become compulsary with MariaDB jdbc driver that permits to see all the schemas inside the MySQL instance).

In all, the string should end up looking something like this: jdbc:mysql://www.someplace.com:3306/mydb

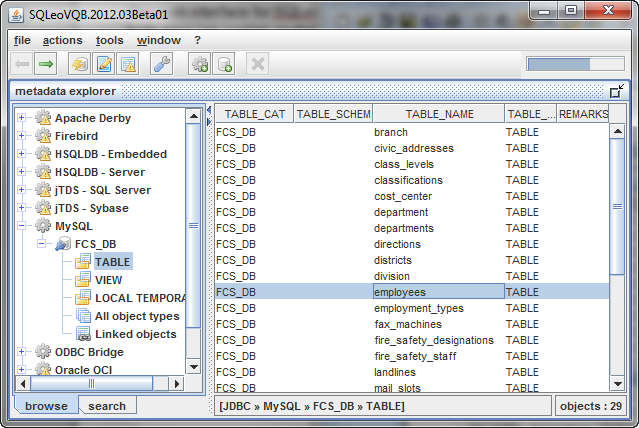
Enter the user name and password assigned to you by the database administrator and check off the additional options as required. Click the OK button to connect to the database.

# Metadata explorer

We can navigate back to the Metadata Explorer by clicking on the button with the image C:\SQLeoVQB.2012.03Beta01\src\images\database_lightning.png and labelled metadata explorer on the button toolbar

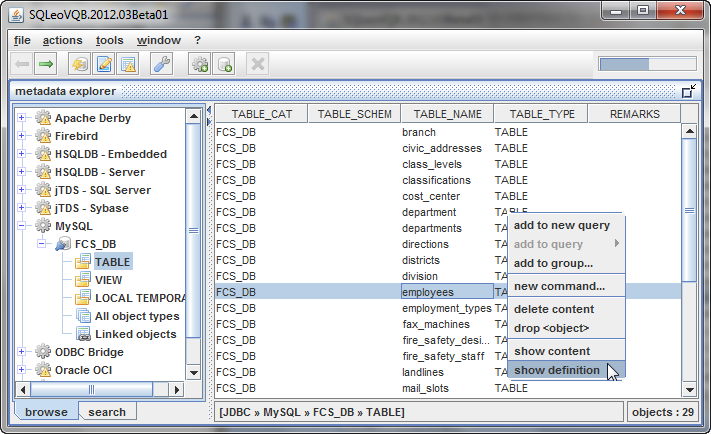
## The Metadata browser mode

The tree structure you see in the left hand pane in IMAGE 3 contains a listing of many of the database types SQLeo can connect to. Using the example below, we can see that a connection has been made to a MySQL database whose name is FCS\_DB. Since the MySQL node in the tree is the active node with a connection, you can open the node further by clicking on the + and – symbol to the left of each node. The FCS\_DB database contains several other entities: a TABLE node, a VIEW node, a LOCAL TEMPORARY node, an ALL OBJECT TYPES node, and a LINKED OBJECTS node. As you select each node the content pane on the right will display different information. In the image, the TABLE node is selected and therefore the content pane will display the list of Tables contained within this database. Currently selected is the *employees* table.

IMAGE 3 

From the Content pane it is possible to drill down even further to discover the details about each table. With the *employees* table already selected, you can use the right mouse button to bring up additional menu items which provide options on the selected table.

IMAGE 4

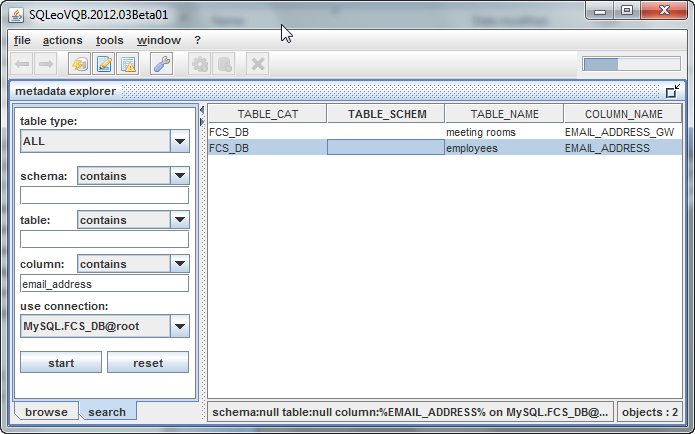


## The Metadata Search Feature

At the bottom of the Metadata Explorer internal window are two tabs. By default the browse tab is selected allowing the user to navigate the tree of database types. Since a database can contain many Tables and each Table can contain many Columns, often a person needs another way to locate certain entities within the database. If we select the **search** tab, we are presented with a number of fields and drop down options that allow us to perform a search on the entire database schema

As an introduction and an example, we have entered the Column named *email\_address* into the field labelled column and have left the default option *contains* selected. When we run the search on our criteria we have the results which you can see in IMAGE 12.

IMAGE 12



Right clicking on the Table gives the same options as sawn in Metadata Explorer browse mode.

# Command Editor

To open the Command Editor window, click on the Command Editor button C:\SQLeoVQB.2012.03Beta01\src\images\page_edit.png on the toolbar.

The Command Editor is a separate internal window you can use instead of the Visual Query Designer. The Command Editor is intended for those who are confident in their abilities to type SQL queries with a thorough understanding of the SQL language. The Command Editor is a great place to practice your skills. If you get your statement wrong, the Command Editor will tell you this is the case. Using **many SQL statements (SELECT, DML, DDL, PL/SQL) in the same window (when query builder only supports ONE SELECT only)**

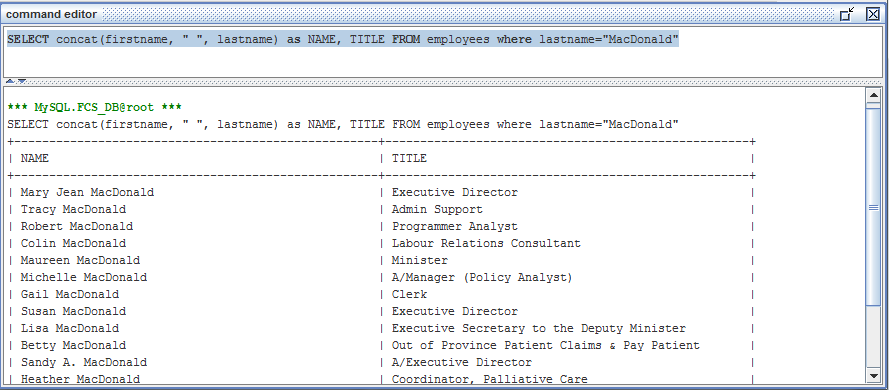
**Permits to Switch between databases connections**

While you can use the Command Editor to perform SELECT queries, it is also used to perform other functions using statements such as ALTER, INSERT, DELETE, UPDATE, etc. To

## Using command editor result in text mode

Each RDBMS generally come with their own set of Functions which you can use in your SELECT statements to help display your results in a way that is meaningful to you. Among these Functions will be: CONCAT, COUNT, YEAR, FLOOR, MAX, MIN, MONTH, NOW, SUM, etc. While we will not cover all of these, we will show you how to use the CONCAT Function in the Command Editor.

IMAGE 34



As you can see from IMAGE 34, the Command Editor has a split pane. Inside the top pane you type your query. When you click on the *launch query* button C:\SQLeoVQB.2012.03Beta01\src\images\table_gear.png on the toolbar, the results of your query will display in the lower pane.

## Using command editor result in grid mode

**Note: starting with 2014.11 version, SELECT results are displayed in data grid format, to get the previous “text” format you can uncheck that option on the tool bar**

## New command

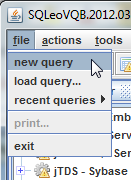
menu: voir Advanced

There is an helper to generate syntax for INSERT, UPDATE and DELETE statements ...

# The Visual Query Designer

The Visual Query Designer is another internal window we can bring up by selecting the File/New Query menu.

IMAGE 15



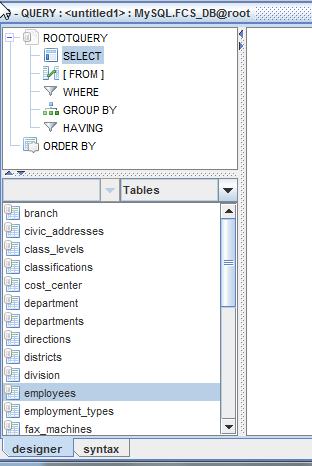
## The Designer Mode

You should see an internal window that appears as in IMAGE 16. Take note that at the bottom of the window are two tabs labelled **designer** and **syntax**. The QUERY window, by default, opens in the designer mode. We will discuss the syntax mode later. The QUERY window will automatically load all the Table names in the lower portion of the window (this can be modified in the preferences). In the top portion you’ll see another tree like structure where each node is labelled in accordance with the known keywords from the SQL language: SELECT, FROM, WHERE, GROUP BY, HAVING, AND ORDER BY. With the use of the right mouse button you will be able to access popup menus that provide additional options when you select each node in the tree.

First, we want to select a Table from which we want to extract data. To construct a simple query, we will select the *employees* Table. You can either double click the table in the list, or you can drag ‘n drop the Table into the Content pane on the right.

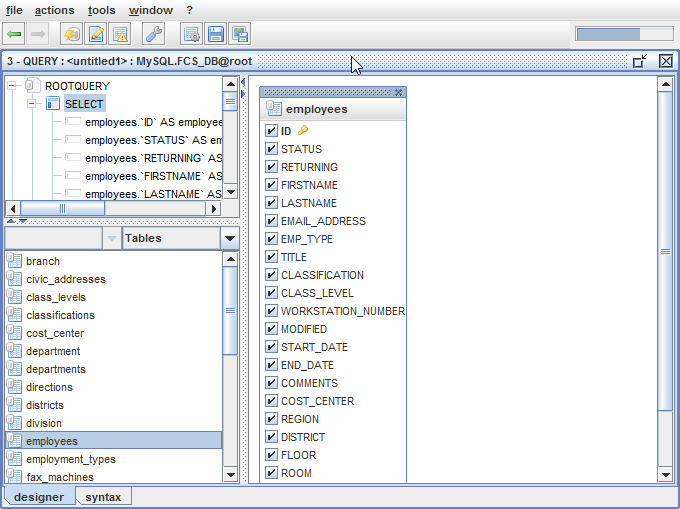
You can perform the same procedure in the Metadata Explorer by selecting a Table, right clicking on it to obtain the popup menus with the options: ***new query*** or ***add to query…***

IMAGE 16



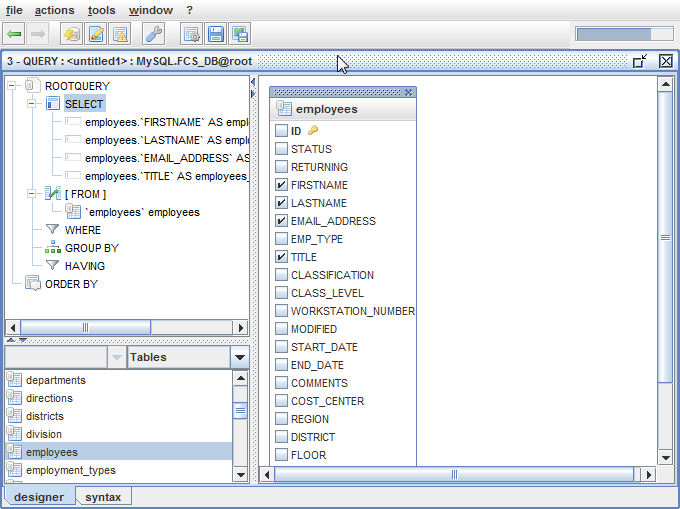
You should see your Table in the Content pane as in IMAGE 17.

IMAGE 17



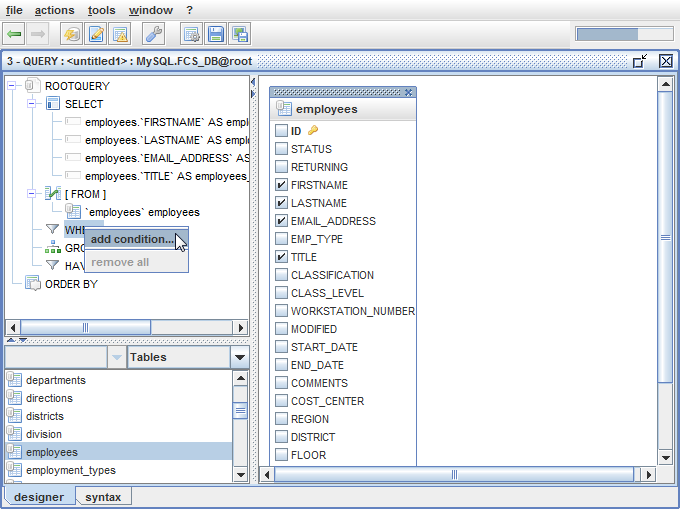
The Content pane displays another internal window that contains the listing of all the Columns contained within the *employees* Table. Each Column name has next to it a checkbox with a check inside each of these. Notice also, that the SELECT node also displays all the Column names. What we need to decide now is what Columns are we really interested in. We will uncheck all the Columns except for FIRSTNAME, LASTNAME, EMAIL\_ADDRESS, and TITLE. As we uncheck the Columns, the list will decrease on the SELECT node. In our example we see the following in IMAGE 18:

IMAGE 18

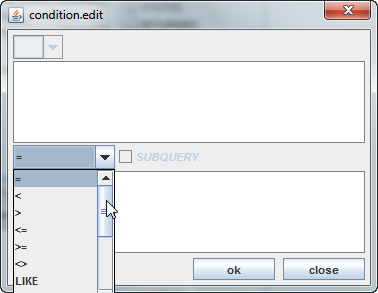


If we were to run this query as is, we would have a tremendous listing of employee names. We are really only interested in seeing records on employees whose last name is “Campbell”. So, we will place a condition on this query stating just that. To add a condition on a query, we will right click on the WHERE node. This will cause a popup menu to appear with the option: add condition.

IMAGE 19

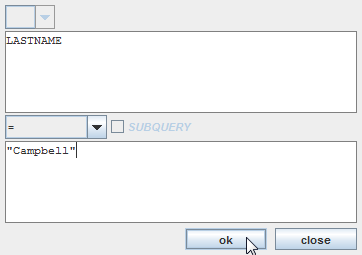


Selecting this option will bring up a dialog box that allows you to create a condition based on several expression operators: =, <, >, <=, >=, <>, LIKE, NOT LIKE, etc. These operators can be accessed from the drop down combo box as seen in IMAGE 20.

IMAGE 20 

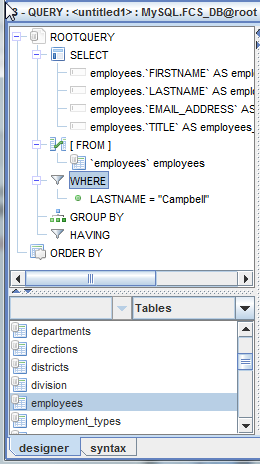
For our example we will leave the default equals sign (=) and type in our condition. See IMAGE 21. The top textbox is where you type the Column name you wish to place the condition on. The bottom textbox is where you type the remainder of the expression. In our case, we type “Campbell” in quotes.

IMAGE 21



After clicking the OK button, we are returned to the QUERY window. Notice now in IMAGE 22 that the WHERE node contains our condition.

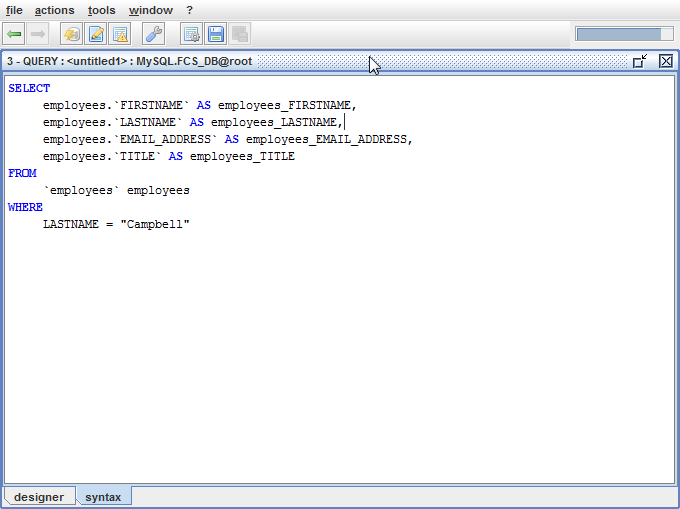
IMAGE 22



## The Syntax Mode

We are almost ready to run our query to see the results, but before we do, please note the two tabs at the bottom of the QUERY window labelled: **designer** and **syntax**. Up to this point we have been in the designer mode of the QUERY window. If we take this time to select the syntax tab, we will be able to see how the actual SQL query has been constructed by SQLeo. See IMAGE 23. This will be the query that is sent to the database and it in turn, will respond with a RecordSet containing tabular data which we can display.

IMAGE 23

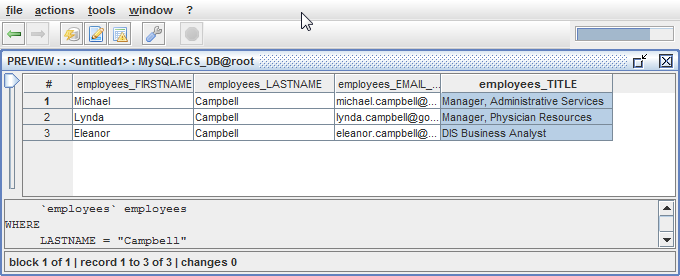


At this point we can run our query by clicking on the button with the image C:\SQLeoVQB.2012.03Beta01\src\images\table_gear.png and labelled *launch query*. For our example, we have the results displayed in the PREVIEW window as in IMAGE 24.

## The PREVIEW Window and Query Results

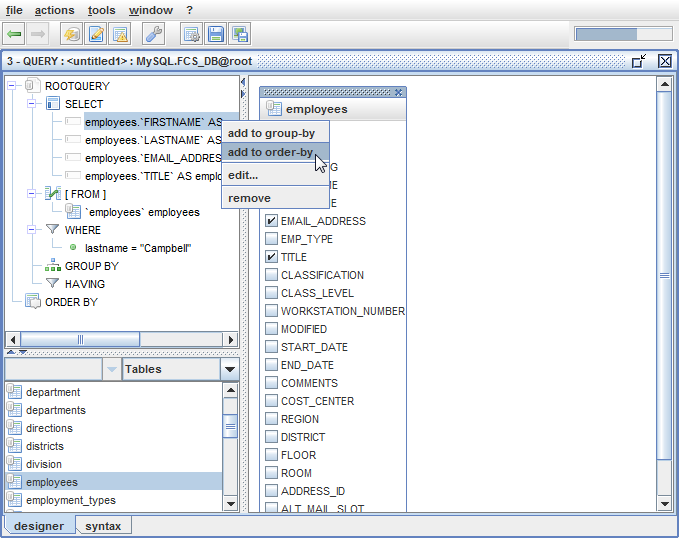
* Supprimées ?

IMAGE 24



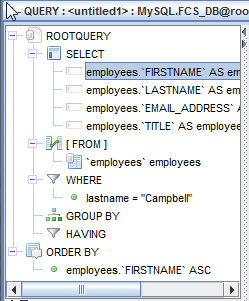
It may be you would want to rearrange the order of your results especially if you have a lengthy list of records. We only have three in our results, but we will re-query the database to give us our results in alphabetical order based on the *FIRSTNAME* Column. To do this, we return to the QUERY window in designer mode as in IMAGE 25. From the tree under the SELECT node, we right click the child node labelled *employees. FIRSTNAME.* A popup menu displays more options. We select ***add to order-by****.*

IMAGE 25



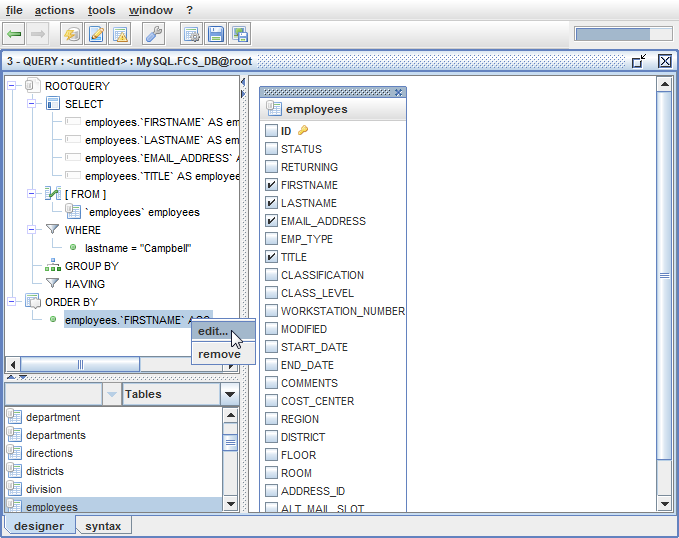
Notice what happens in IMAGE 26. The ORDER BY node now contains the new condition.

IMAGE 26



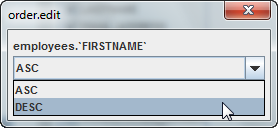
If you look closer, you can see that the ORDER BY condition by default will arrange the resulting records in ascending order. The suffix ASC is displayed. You may have an occasion when you would prefer to display your records in descending order. You can change this if you wish. Right click on the condition as it appears under the ORDER BY node and select the *edit…* option as in IMAGE 27.

IMAGE 27



This will launch a dialog box as in IMAGE 28. From here you can select the descending order option.

IMAGE 28



We will not change the order of the results. Instead we will leave the option to display in ascending order. If we now launch the query by clicking on the *launch query* button C:\SQLeoVQB.2012.03Beta01\src\images\table_gear.png, we will see that our resulting records are now in alphabetical order by first name. See IMAGE 29.

IMAGE 29

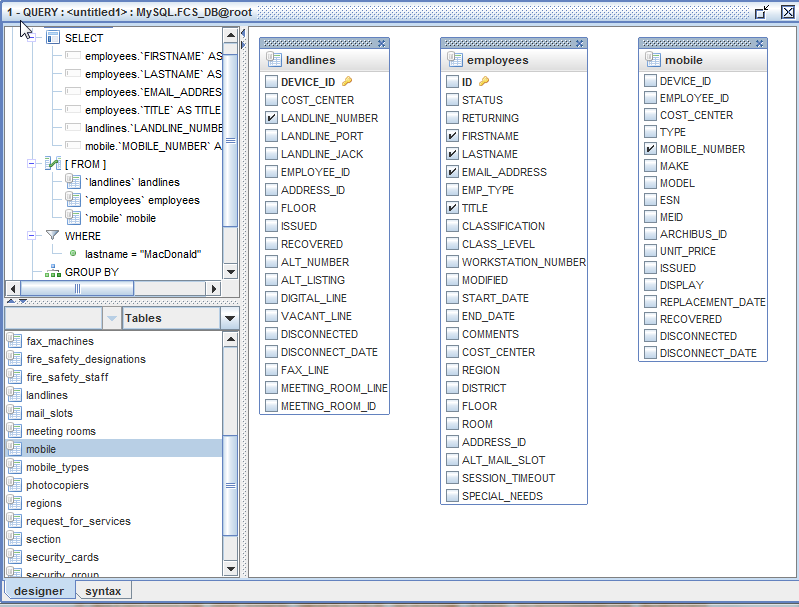


At this point you have the option to save your query to a file so that you can recall it later, instead of having to duplicate all the steps it has taken you to create it. Return to the QUERY window using the back button C:\SQLeoVQB.2012.03Beta01\src\images\arrow_left.png. Look for the Save Query button C:\SQLeoVQB.2012.03Beta01\src\images\disk.png and click on it. Step through the Save As dialog wizard; provide a name for your query file and save it to a directory of your choosing. By default query files are generally saved using the .sql extension. SQLeo has its own file type which uses the .xlq extension. You may choose either one.

## Adding More Tables To A Query

The time has come to perform a more complex query. So far we have been working with the *employees* Table. Now we are going to include columns from two other Tables: LANDLINES and MOBILE. Our database separates information regarding landline phones and mobile phones. If we wish to include the phone number and mobile phone number for each of the employees, we are going to need to include the LANDLINES and MOBILE Tables to the Content pane in the QUERY window. To do this, we either double-click those tables as they are listed or drag ‘n drop them into the Content pane. When we have completed this task, we will have all three Tables displayed in the Content pane. See IMAGE 30.

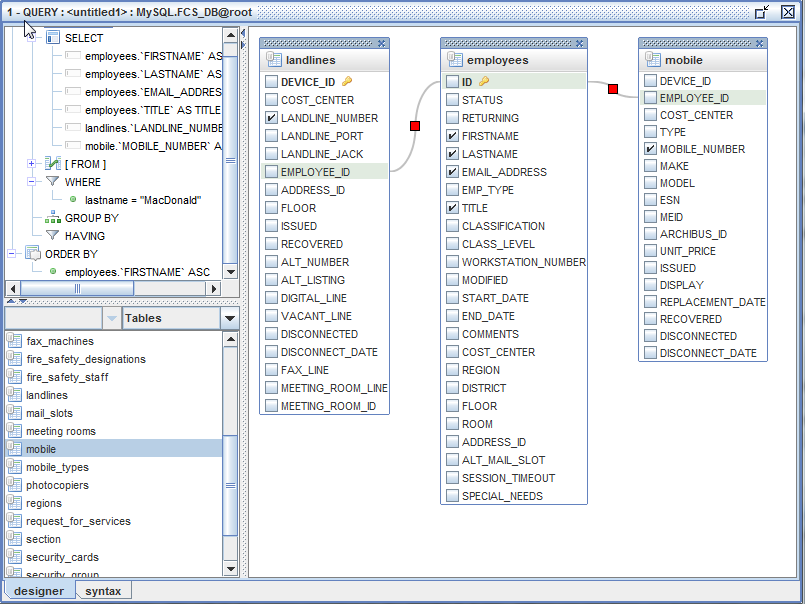
IMAGE 30



Notice that we have deselected every column from the LANDLINES Table except for the LANDLINE\_NUMBER column. Also, we have deselected every column from the MOBILE Table except for the MOBILE\_NUMBER column. These two columns are all we are interested in adding to our query. Note that we have also changed the criteria on the WHERE clause to: lastname=”MacDonald”.

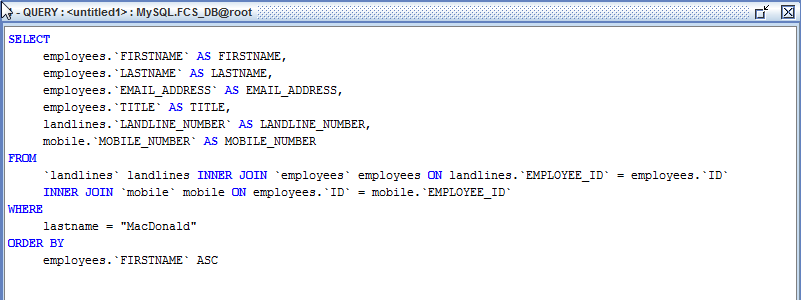
At this point we need to JOIN our Tables together by performing a little drag ‘n drop operation. Using the left mouse button we will drag ‘n drop the EMPLOYEES **ID** column to the **EMPLOYEE\_ID** column on the LANDLINES Table. We will do the same operation to the MOBILE Table. When we are done, we have two links joining the three Tables as seen in IMAGE 31.

IMAGE 31



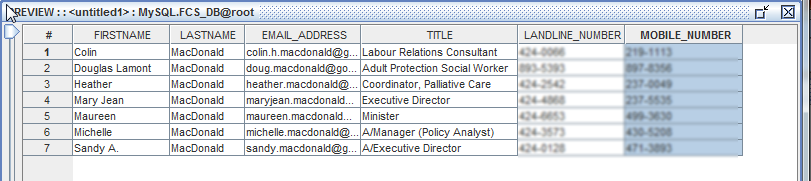
The EMPLOYEE\_ID columns in both the LANDLINES and MOBILE Tables contain integer values referencing every employee record in the EMPLOYEES Table. The database will attempt to match these values up when it runs the SELECT query. To see how this looks as an SQL query, we click on the **syntax** tab. See IMAGE 32.

IMAGE 32



Now we will launch the query by clicking on the *launch query* button C:\SQLeoVQB.2012.03Beta01\src\images\table_gear.png. See IMAGE 33 for the results.

IMAGE 33



Our query results in seven records. The values for the LANDLINE\_NUMBER and MOBILE\_NUMBER have been blurred to protect the privacy of the individuals listed.

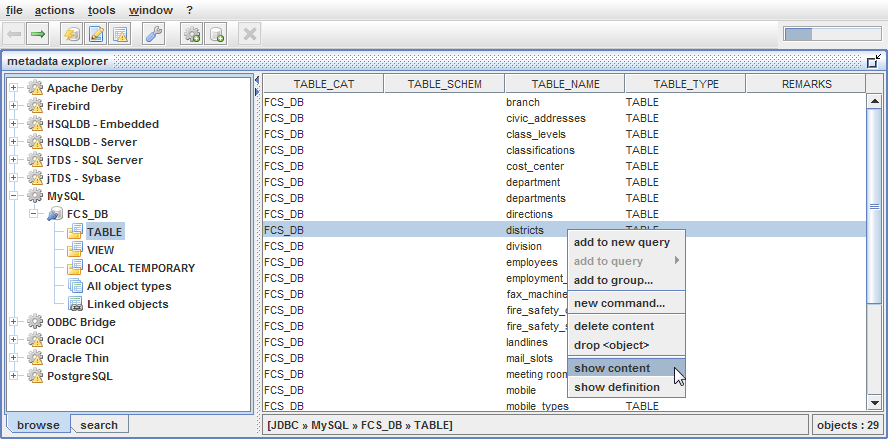
Let us return to IMAGE 31. Note that the JOIN lines connecting the Tables contain a RED square in the center. At any time you may edit the JOINs by using your right mouse button and clicking on the RED square. You will be provided with two menu options: EDIT or REMOVE. If you choose EDIT, you will be presented with a dialog box that allows you to make some refinements on the JOIN criteria.

# The Content Window

Let us take a step back, and return to the Content Window since there is more to explore there. We navigate to the Content Window first by returning to the Metadata Explorer. From there we can choose any Table being displayed in the Content pane, and using double click or the right mouse button, we select *show content* from the menu options on the Table named *districts*.

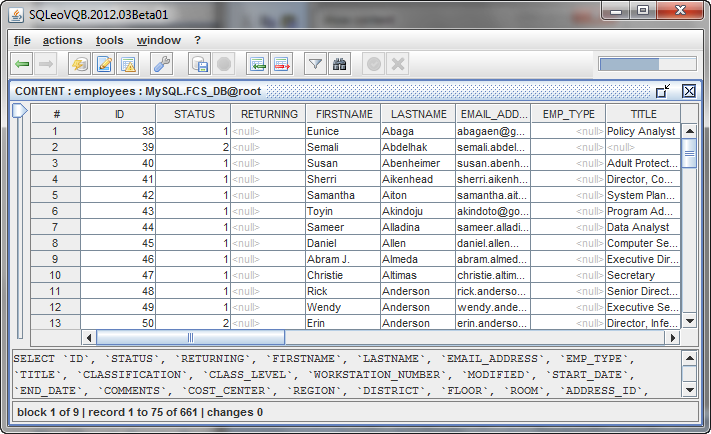
## Displaying data of a Table (or a query)

IMAGE 36



When the Content Window is displayed

IMAGE 10



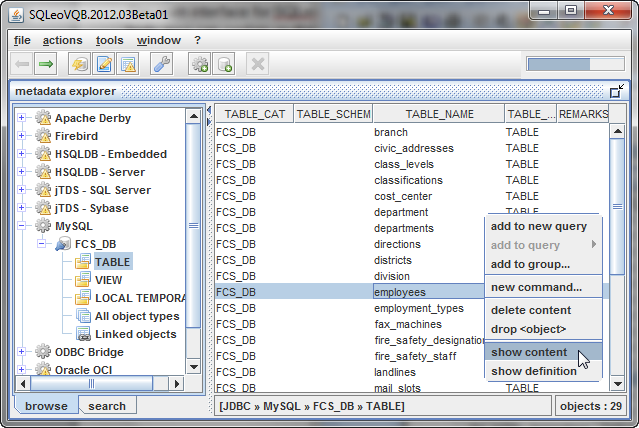
At the bottom of the CONTENT window is the current SQL query.

This Content window will not load all the records in memory, but only the first 100 records (this is a parameter that can be modified in preferences). As the user scrolls down the lists of records SQLeo will retrieve the next 100 records for display.

## Sorting Data

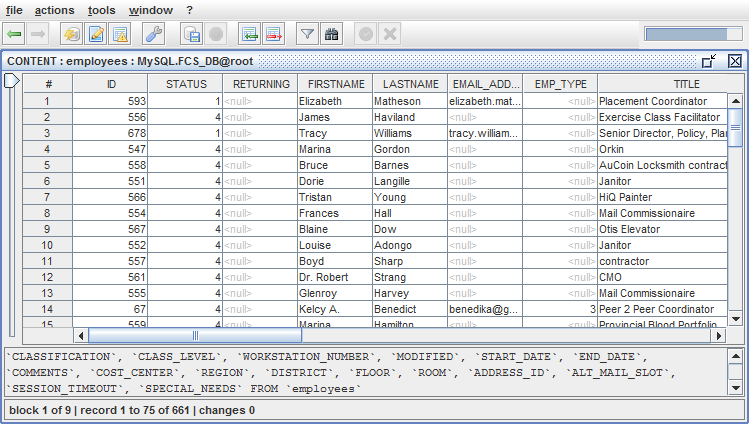
When dealing with Tables that contain a lot of data, a sorting feature can make that data a lot easier to work with. Let us return to the Metadata Explorer and work with the *employees* Table again. We want to open up all the data in this Table, so using the right mouse button, we will double click or right click on the *employees* Table and select the *show contents…* menu. See IMAGE 45.

IMAGE 45



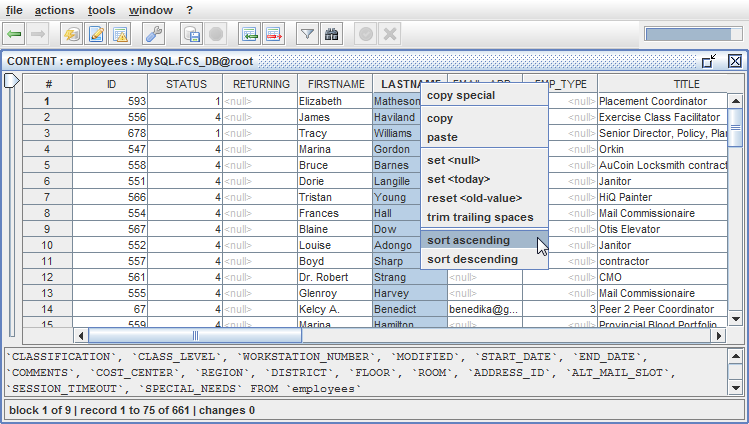
When our Content window opens, our display looks as it does in IMAGE 46.

IMAGE 46



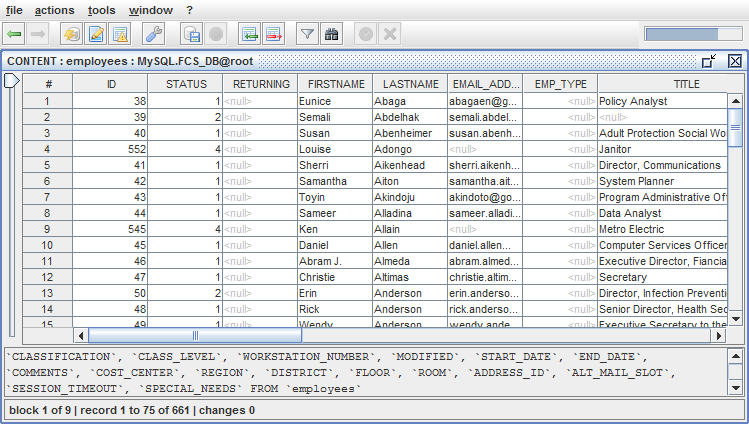
From what we can see in the image, the data is in no particular order. If you want to start making sense of it all, it may be helpful to sort the data by *employees.LASTNAME*. To perform a sort on the LASTNAME column, we use the right mouse button on the column header and select *sort ascending…* from the list of menu items as in IMAGE 47.

IMAGE 47



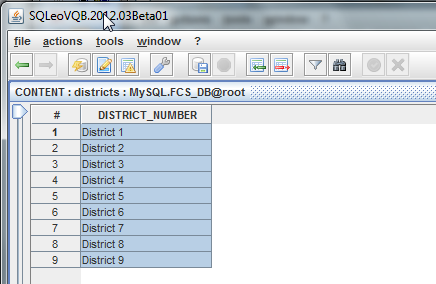
The result of the sort appears as in IMAGE 48. All the last names beginning with the letter A appear at the top of our display. Of course you have the option to sort in descending order. If your review IMAGE 47, you can see that option is available among the menu items.

IMAGE 48



The toolbar has a specific set of buttons:

The *insert record* button C:\SQLeoVQB.2012.03Beta01\src\images\table_row_insert.png will allow you to insert a new record, while the *delete record* button C:\SQLeoVQB.2012.03Beta01\src\images\table_row_delete.png allows you to remove any record in the Table. The *filter* C:\SQLeoVQB.2012.03Beta01\src\images\filter.png and *find* C:\SQLeoVQB.2012.03Beta01\src\images\find.png buttons we will discuss later.

IMAGE 37 

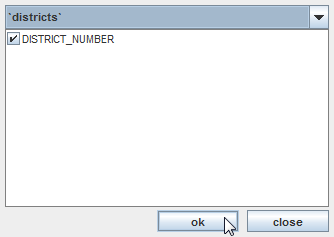
## Inserting and Deleting Records from a Table

In this window, it not only displays the data in the *districts* Table, but it also allows you to make changes to the Table. This is assuming you have the appropriate RIGHTS or privileges to make those changes in accordance to your user account on the database system. We have complete access to our database so we can show you some of the possible changes you can make to the Table from this window. The fields under the Column named *DISTRICT\_NUMBER* can be edited. The values displayed under the Column labelled with the hash symbol # cannot be changed.

Before we can implement any changes to this Table we need to tell SQLeo on which Columns we want to make changes. In this case, there is only one Column. For this step, we select the *actions* menu and choose the submenu labelled *update criteria…* This will launch a dialog box as in IMAGE 38.

Note: If table has a Primary Key, its fields will be used by default for updates, if not *update criteria* has to be defined manually to tell the tool how to perform the data modifcation

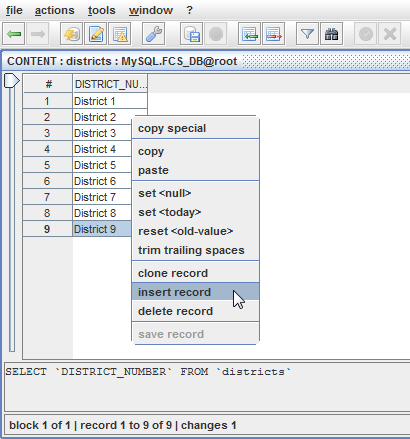
IMAGE 38



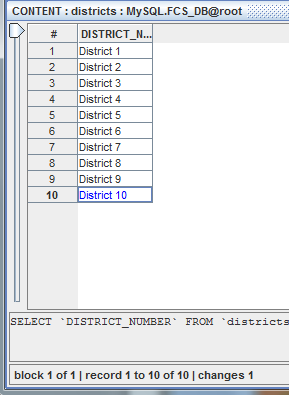
We place a check in the checkbox next to the Column named *DISTRICT\_NUMBER* and click the OK button. At this point, if we implement any changes under the Column *DISTRICT\_NUMBER* the changes will be recorded in the form of SQL statements. Before we commit the changes to the database Table, we will have a look at the SQL statements that were recorded. But first, let us make some changes.

We will select the ninth record. We will insert a new record just below it. We can either click on the *insert record* C:\SQLeoVQB.2012.03Beta01\src\images\table_row_insert.pngbutton, or we can use the right mouse button to display popup menus. One happens to be an *insert record* menu item. See IMAGE 39.

IMAGE 39

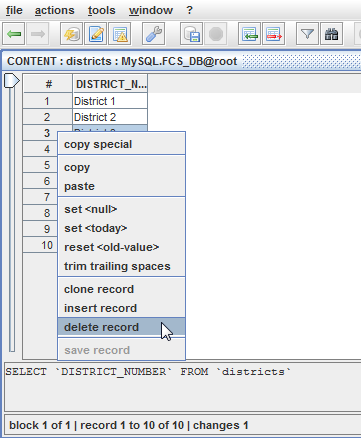


Take this time to note the other menu item options that are available for future reference. Notice there is a handy menu item labelled *reset <old-value>.* This has the same ability as an undo feature to return a prior value from a cell or field. After selecting the *insert record* menu, we obtain a new record item which is automatically given an ID value of 10. In the blank cell or field we type District 10. See IMAGE 40. Note that any changes made show in a blue font to indicate the change has **not** been saved yet.

IMAGE 40 

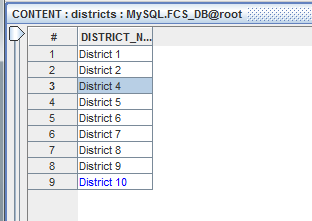
Before we save our changes, let us remove a record from this Table. District 3 is no longer required, so we will remove it. We right click on record three and select the *delete record* menu. See IMAGE 41.

IMAGE 41



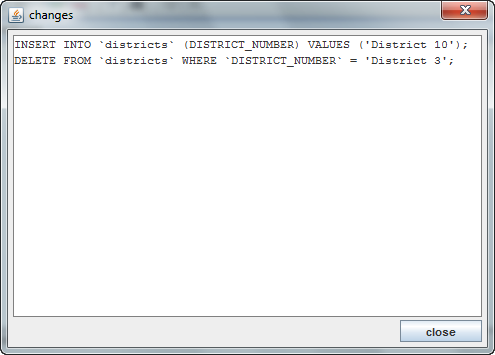
Right away the record containing District 3 is removed as in IMAGE 42.

IMAGE 42



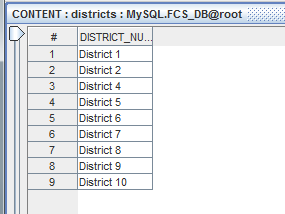
At this point we can have a look at the SQL statements that will be applied to the database before we save our changes. We select the *actions* menu and select the *show changes…* submenu. This brings up a dialog box as in IMAGE 43.

IMAGE 43



This is a good opportunity to look closely at the syntax for inserting and deleting records from a Table using the SQL language. We click on the Close button and now we can save our changes to the database by clicking on the *apply changes to db C:\SQLeoVQB.2012.03Beta01\src\images\database_save.png* button. Once the changes are saved, any field that was edited and in blue font will now display in a normal black font. See IMAGE 44.

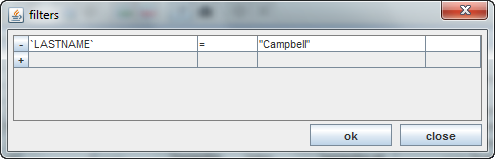
IMAGE 44



## Filtering Data

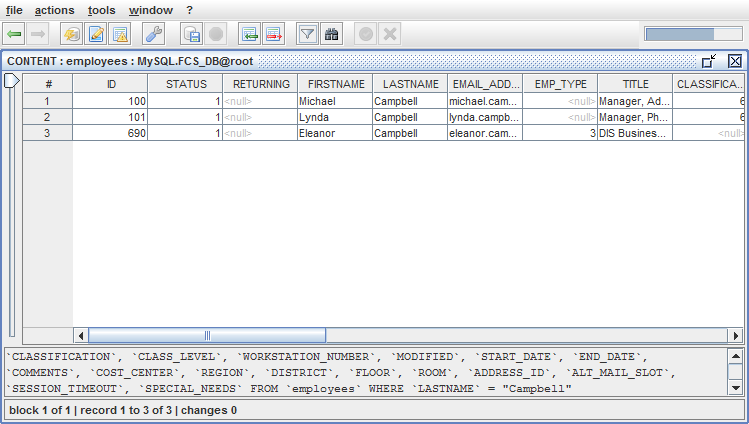
If you wish to narrow your focus on certain records, you can filter the data to only those records that are of interest. From the toolbar we will select the filter C:\SQLeoVQB.2012.03Beta01\src\images\filter.pngbutton. Doing this brings up a dialog box where we can enter criteria. We will propose a simple criteria and enter LASTNAME = “Campbell” as in IMAGE 49.

IMAGE 49



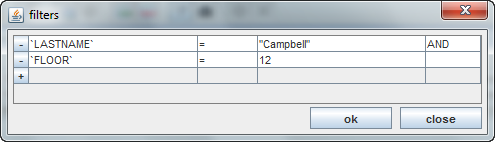
When we click on the OK button, the CONTENT window displays only those records whose LASTNAME equals “Campbell”. See IMAGE 50.

IMAGE 50



Returning to the filters dialog box, note that there are numerous options to set your criteria. Where the equals symbol is displayed in IMAGE 49, you will find the options: =, <, >, <=, >=, <>, LIKE, NOT LIKE, etc. You can also enter more than one criteria using the AND keyword, or OR keyword. See IMAGE 51.

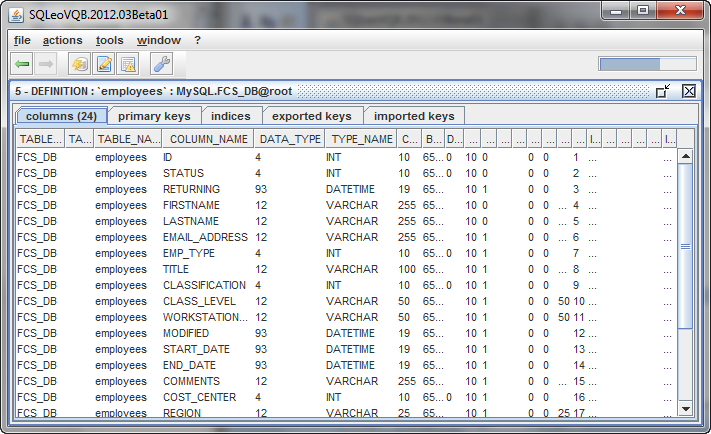
IMAGE 51



## Finding Terms

One final search option is available to the user. It is the Find C:\SQLeoVQB.2012.03Beta01\src\images\find.png button. Clicking on this button on the toolbar brings up a familiar find dialog box which you can use to find, and even replace, any term that may reside inside all the data within a given Table. This feature is typically found in many other applications and when you tell it to perform its search; it will highlight the cell in a yellow color.

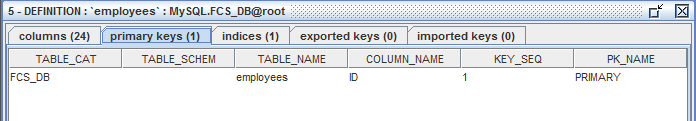
# The DEFINITION Window

IMAGE 5

From IMAGE 5 you can see that the DEFINITION internal window displays details on the *employees* Table providing such information as Column names and their data types: INT, DATETIME, VARCHAR, etc.

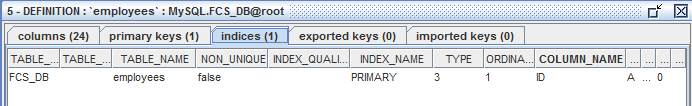
You will notice that the DEFINITION internal window has a number of tabs you can select to obtain other information on the *employees* Table. If we select the *primary keys* tab, we can discover which Column has been set as having a Primary Key. In this case, as seen in IMAGE 6, the Column named ID has the Primary Key. Typically one Primary Key is assigned to each Table by the database designer, however not all Tables will necessarily have a Primary Key.

IMAGE 6



If we select the *indices* tab as in IMAGE 7, we can see that the Column ID has not only a Primary Key, but also is indexed to allow for faster searches when the database performs a query on this particular Table. The *exported keys* and *imported keys* tabs provide additional information regarding Reference Primary Keys and Foreign Keys respectively. Not every database system supports these features, which is why you will notice the value zero indicated on the tabs.

IMAGE 7



# SQL History

To be described

# Data comparer

To be described

# Schema comparer

To be described

# Troubleshooting

Log files ?

# Support

At the time of this writing support and discussion of issues can be obtained from the SourceForge website: <http://sourceforge.net/p/sqleo/discussion/>