Query Design

The **Query Design View** allows you to create and edit a database query.

Some databases also support the creation of a new table view. Selecting the **Create View** command from the **Tables** tab page of a database document, you see the **View Design** window that resembles the **Query Design** window described here.

To access this command...
In a database file window, click the Queries icon, then choose Edit - Edit

The Design View

To create a query, click the **Queries** icon in a database document, then click **Create Query in Design View**.

The lower pane of the Design View is where you define the query. To define a query, specify the database field names to include and the criteria for displaying the fields. To rearrange the columns in the lower pane of the Design View, drag a column header to a new location, or select the column and press Ctrl+arrow key.

In the top of the query Design View window, the icons of the **Query Design** Bar and the **Design** bar are displayed.

If you want to test a query, double-click the query name in the database document. The query result is displayed in a table similar to the Data Source View. Note: the table displayed is only temporary.

Browse

When you open the query design for the first time, in order to create a new query, you can click **Add Tables**. You then see a dialog in which you must first select the table that will be the basis for the query.

Double-click fields to add them to the query. Drag-and-drop to define relations.

While designing a query, you cannot modify the selected tables.

Remove tables

To remove the table from Design View, click the upper border of the table window and display the context menu. You can use the **Delete** command to remove the table from the Design View. Another option is to press the Delete key.

Move table and modify table size

You can resize and arrange the tables according to your preferences. To move tables, drag the upper border to the desired position. Enlarge or reduce the size in which the table is displayed by positioning the mouse cursor on a border or on a corner and dragging the table until it is the desired size.

Table Relations

If there are data relations between a field name in one table and a field name in another table, you can use these relations for your query.

If, for example, you have a spreadsheet for articles identified by an article number, and a spreadsheet for customers in which you record all articles that a customer orders using the corresponding article numbers, then there is a relationship between the two "article number" data fields. If you now want to create a query that returns all articles that a customer has ordered, you must retrieve data from two spreadsheets. To do this, you must tell OpenOffice.org what the relationship exists between the data in the two spreadsheets.

To do this, click a field name in a table (for example, the field name "Item-Number" from the Customer table), hold down the mouse button and then drag the field name to the field name of the other table ("Item-Number" from the Item table). When you release the mouse button, a line connecting the two fields in the two windows appears. The corresponding condition that the content of the two field names must be identical is entered in the resulting SQL query.

The creation of a query that is based on several related sheets is only possible if you use OpenOffice.org as the interface for a relational database.

You cannot access tables from different databases in a query. Queries involving multiple tables can only be created within one database.

Specifying link type

If you double-click the line connecting two linked fields or call the menu command **Insert - New Relation**, you can specify the type of link in the **Relations** dialog.

Alternatively, press Tab until the joint vector is selected (it is displayed enlarged), then press Shift+F10 to display the context menu and there choose the command **Edit**.

Deleting relations

To delete a relation between two tables, click the connection line and then press the Delete key.

Alternatively, delete the respective entries in **Fields involved** in the **Relations** dialog. Or press Tab until the connecting vector is displayed highlighted, then press Shift+F10 to open the context menu and select **Delete** command.

Define query

Select conditions to define the query. Each column of the design table accepts a data field for the query. The conditions in one row are linked with a Boolean AND.

Specify field name

First, select all field names from the tables that you want to add to the query. You can do this either with drag-and-drop or by double-clicking a field name in the table window. With the drag-and-drop method, use the mouse to drag a field name from the table window into the lower area of the query design. As you do this, you can decide which column you want to add the field to. Select a field name by double-clicking. It will then be added to the next free column.

Deleting field names

To remove a field name from the query, click the column header of the field and choose the **Delete** command on the context menu for the column.

Save query

Use the **Save** icon on the Standard Bar to save the query. You see a dialog that asks you to enter a name for the query. If the database supports schemas, you can also enter a schema.

Schema

Enter the name of the schema that is assigned to the query or table view.

Query name or table view name

Enter the name of the query or table view.

Filtering data

To filter data for the query, set the desired preferences in the lower area of the Design View. The following lines are available:

Field

Enter the name of the data field that you referred to in the Query. All settings made in the lower rows refer to this field. If you activate a cell with a mouse click you'll see an arrow button, which

enables you to select a field. The "Table name.*" option selects all data fields and the criteria is valid for all table fields.

Alias

Specifies an alias. This alias will be listed in a query instead of the field name. This makes it possible to use user-defined column labels. For example, if the data field has the name PtNo and, instead of that name, you would like to have PartNum appear in the query, enter PartNum as alias.

In an SQL statement, aliases are defined as following:

SELECT column AS alias FROM table.

For example:

SELECT "PtNo" AS "PartNum" FROM "Parts"

Table

The corresponding database table of the selected data field is listed here. If you activate the a cell with a mouse click, an arrow will appear which enables you to select another table of the current query.

Sort

If you click the cell, you can select among the sorting options: ascending, descending and not sorted. Text fields will be sorted alphabetically (A to Z) and numerical fields numerically (0 to 9).

Visible

If you mark the **Visible** property for a data field, that field will be visible in the query. If you only use a data field to formulate a condition, you do not necessarily need to show it.

Criteria

Specifies the criteria by which the content of the data field should be filtered.

or

Here you can enter one additional criterion for filtering in each line. Multiple criteria in one column will be connected by an OR link.

You can also use the context menu of the line headers in the lower area of the query design to insert another line for functions:

Functions

You can also enter function calls directly into the SQL statement. The syntax is:

SELECT FUNCTION(column) FROM table.

For example, the function call in SQL for calculating a sum is:

SELECT SUM("Price") FROM "Article".

Except for the **Group** function, the above functions are so-called Aggregate functions. These are functions that calculate data to create summaries from the results. Additional functions that are not listed in the list box are also possible. These depend on the specific database system in use. To get information about driver specific functions refer to the documentation of your database system.

To use other functions not listed in the list box, you must enter them under **Field**. They will then appear automatically in the **Function** line.

You can also assign aliases to function calls. If the query is not to be displayed in the column header, enter the desired name under **Alias**.

The corresponding function in an SQL statement is:

SELECT FUNCTION() AS alias FROM table

Example:

SELECT COUNT(*) AS count FROM "Item"

If you run this function, you cannot insert any additional columns for the query other than receiving these columns as a "Group" function.

Examples

In the following example, a query is run through two tables: an "Item" table with the "Item_No" field and a "Suppliers" table with the "Supplier_Name" field. In addition, both tables have a common field name "Supplier_No."

The following steps are required to create a query containing all suppliers who deliver more than three items.

1. Insert the "Item" and "Suppliers" tables into the query design.
2. Link the "Supplier_No" fields of the two tables if there is not already a relation of this type.
3. Double-click the "Item_No" field from the "Item" table. Display the **Function** line using the context menu and select the Count function.
4. Enter >3 as a criterion and disable the Visible field.
5. Double-click the "Supplier_Name" field in the "Suppliers" table and choose the Group function.
6. Run the query.

If the "price" (for the individual price of an article) and "Supplier_No" (for the supplier of the

article) fields exist in the "Item" table, you can obtain the average price of the item that a supplier provides with the following query:

1. Insert the "Item" table into the query design.
2. Double-click the "Price" and "Supplier_No" fields.
3. Enable the **Function** line and select the Average function from the "Price" field.
4. You can also enter "Average" in the line for the alias name (without quotation marks).
5. Choose Group for the "Supplier_No" field.
6. Run the query.

The following context menu commands and symbols are available:

Functions

Shows or hides a row for selection of functions.

Table Name

Shows or hides the row for the table name.

Alias Name

Shows or hides the row for the alias name.

Distinct Values

Applies only distinct values to the query. This applies to records containing data that appears several times in the selected fields. If the **Distinct Values** command is active, you will see only one record in the query (DISTINCT). Otherwise, you will see all records corresponding to the query criteria (ALL).

For example, if the name "Smith" occurs several times in your address database, you can choose the **Distinct Values** command to specify in the query that the name "Smith" will occur only once.

For a query involving several fields, the combination of values from all fields must be unique so that the result can be formed from a specific record. For example, you have "Smith in Chicago" once in your address book and "Smith in London" twice. With the **Distinct Values** command, the query will use the two fields "last name" and "city" and return the query result "Smith in Chicago" once and "Smith in London" once.

In SQL, this command corresponds to the DISTINCT predicate.

Formulating filter conditions

When formulating filter conditions, various operators and commands are available to you. Apart

from the relational operators, there are SQL-specific commands that query the content of database fields. If you use these commands in the OpenOffice.org syntax, OpenOffice.org automatically converts these into the corresponding SQL syntax. You can also enter the SQL command directly. The following tables give an overview of the operators and commands:

Operator	**Meaning**	**Condition is satisfied if...**
=	equal to	... the content of the field is identical to the indicated expression. The operator = will not be displayed in the query fields. If you enter a value without any operator, the operator = will be automatically adopted.
<>	not equal to	... the content of the field does not correspond to the specified expression.
>	greater than	... the content of the field is greater than the specified expression.
<	less than	... the content of the field is less than the specified expression.
>=	greater than or equal to	... the content of the field is greater than or equal to the specified expression.
<=	less than or equal to	... the content of the field is less than or equal to the specified expression.

OpenOffice.org command	**SQL command**	**Meaning**	**Condition is satisfied if...**
IS EMPTY	IS NULL	is null	... The field name is empty. For Yes/No fields with three states, this command automatically queries the undetermined state (neither Yes nor No).
IS NOT EMPTY	IS NOT NULL	is not empty	... the field name is not empty.
LIKE (placeholder * for any number of characters placeholder ? for exactly one character)	LIKE (% placeholder for any number of characters Placeholder _ for exactly one character)	is an element of	... the data field contains the indicated expression. The (*) placeholder indicates whether the expression x occurs at the beginning of (x*), at the end of (*x) or inside the field content (*x*). You can enter as a placeholder in SQL queries either the SQL % character or the familiar (*) file system placeholder in the OpenOffice.org interface. The * or % placeholder stands for any number of characters. The question mark (?) in the OpenOffice.org interface or the underscore (_) in SQL queries is used to represent exactly one character.
NOT LIKE	NOT LIKE	Is not an element of	... the field name does not contain the specified expression.
BETWEEN x AND y	BETWEEN x AND y	falls within the interval [x,y]	... the field name contains a value that lies between the two values x and y.
NOT BETWEEN x AND y	NOT BETWEEN x AND y	Does not fall within the interval [x,y]	... the field name contains a value that does not lie between the two values x and y.
IN (a; b; c...) Note that the	IN (a, b, c...)	contains a, b, c...	... the field name contains one of the specified expressions a, b, c,... Any number of expressions can be specified, and

semicolons are used as separators in all value lists!			the result of the query is determined by an Or link. The expressions a, b, c... can be either numbers or characters
NOT IN (a; b; c...)	NOT IN (a, b, c...)	does not contain a, b, c...	... the field name does not contain one of the specified expressions a, b, c,...
= TRUE	= TRUE	has the value True	... the field name has the value True.
= FALSE	= FALSE	has the value false	... the field name has the value false.

Examples

='Ms.'	returns field names with the field content "Ms."
LIKE 'g?ve'	returns field names with field content such as "give" and "gave".
LIKE 'S*'	returns data fields with field contents such as "Sun".
BETWEEN 10 AND 20	returns field names with field content between the values 10 and 20. (The fields can be either text fields or number fields).
IN (1; 3; 5; 7)	returns field names with the values 1, 3, 5, 7. If the field name contains an item number, for example, you can create a query that returns the item having the specified number.
NOT IN ('Smith')	returns field names that do not contain "Smith".

Date fields are represented as #Date# to clearly identify them as dates. The date condition will be reproduced in the resulting SQL statement in the following ODBC - compliant way:

Date	{D'YYYY-MM-DD'}
Date time	{D'YYYY-MM-DD HH:MM:SS'}
Time	{D'HH:MM:SS'}

OpenOffice.org also supports the following **Escape sequences** known from ODBC and JDBC:

Date	{d 'YYYY-MM-DD'}
Time	{t 'HH:MI:SS[.SS]'} - [ ] optional
DateTime	{ts 'YYYY-MM-DD HH:MI:SS[.SS]'} - [ ] optional

Example: select {d '1999-12-31'} from world.years

Like Escape Sequence: {escape 'escape-character'}

Example: select * from Item where ItemName like 'The *%' {escape '*'}

The example will give you all of the entries where the item name begins with 'The *'. This means that you can also search for characters that would otherwise be interpreted as placeholders, such as *, ?, _, % or the period.

Outer Join Escape Sequence: {oj outer-join}

Example: select Article.* from {oj item LEFT OUTER JOIN orders ON item.no=orders.ANR}

Querying text fields

To query the content of a text field, you must put the expression between single quotes. The distinction between uppercase and lowercase letters depends on the database in use. LIKE, by definition, is case-sensitive (though some databases don't see it that strict).

Querying date fields

Even if you want to filter by a date, you must place the expression between single quotation marks. The following formats are valid: YYYY-MM-DD HH:MM:SS and YYYY/MM/DD HH:MM:SS as well as YYYY.MM.DD HH:MM:SS

Querying Yes/No fields

To query Yes/No fields, use the following syntax for dBase tables:

Status	Query criterion	Example
Yes	for dBase tables: not equal to any given value	=1 returns all records where the Yes/No field has the status "Yes" or "On" (selected in black),
No	.	=0 returns all records for which the Yes/No field has the status "No" or "Off" (no selection).
Null	IS NULL	IS NULL returns all records for which the Yes/No field has neither of the states Yes or No (selected in gray).

The syntax depends on the database system used. You should also note that Yes/No fields can be defined differently (only 2 states instead of 3).

Parameter queries

You must place the variable between square brackets (=[x]) to create a query with variable parameters. Alternatively, you can use an equal sign followed by a colon (=:x). When the query is executed, the program will display a dialog asking you for the expression to which the variable x should be assigned.

If you query several parameters at the same time, you will see a list field in the dialog containing all of the parameters and an input line alongside each one. Enter the values, preferably from top to bottom, and press the Enter key after each line.

Parameter queries with placeholders (*, _) or special characters (for example, ?) are not possible.

If you formulate a parameter query and you save it with the variables, you can later create a query in which only the variables have to be replaced by the expressions that you want. OpenOffice.org asks for these variables in a dialog as soon as you open the query.

Parameter Input

The **Parameter Input** dialog asks you which variables you defined in the query. Enter a value for each query variable and confirm by clicking **OK**.

Parameter queries are also used for subforms, since they work exclusively with queries for which the values to be invoked are read internally from a variable.

A parameter query can have the following form in an SQL statement:

select * from 'addresses' where 'name' = :placeholder

SQL Mode

SQL stands for "Structured Query Language" and describes instructions for updating and administering relational databases.

In OpenOffice.org you do not need any knowledge of SQL for most queries, since you do not have to enter the SQL code. If you create a query in the query design, OpenOffice.org automatically converts your instructions into the corresponding SQL syntax. If, with the help of the **Switch Design View On/Off** button, you change to the SQL view, you can see the SQL commands for a query that has been created previously.

You can formulate your query directly in the SQL code. Note, however, that the special syntax is dependent upon the database system that you use.

If you enter the SQL code manually, you can create SQL-specific queries that are not supported by the graphical interface in **Query design**. These queries must be executed in native SQL mode.

By clicking the **Run SQL command directly** icon in the SQL view, you can formulate a query that is not processed by OpenOffice.org.