

eBase v8
QB and query building Manual



Contents

1.	Methods of extracting data from the system	3
2.	The Query Builder	3
2.1	Creating a new QB query.	3
2.2	Selecting Fields	5
2.3	Preview	5
2.4	Export and save query	6
3.	Creating queries for custom reports	6
3.1	Components of an SQL query	7
3.2	Sample query	8

1. Methods of extracting data from the system

One of the main functions of the eBase is to retrieve and analyze data from the system. Several methods are available in the system for this purpose. This manual describes the two methods available directly from the UI (User Interface). These methods are:

- the Query Builder (QB);
- the Reporting module (RM).

The difference between the two is that the QB can be used primarily to extract "standard" lists from the system. It is relatively easy to use, but it also has some limitations. The RM has a higher learning curve because data extraction requires the use of a separate "query" language. The advantage, however, is that the data can be combined and compiled in more ways.

2. The Query Builder

The QB allows you to extract, combine and export data from several "standard" sources. These "standard" sources include:

- Persons;
- Episodes;
- Treatments;
- Signatures;
- Checklists.

These sources contain data coming from various logical tables and are grouped in a recognizable way. The sources can be combined and filters can be applied to the exports. The relationships between the sources are determined by the QB.

2.1 Creating a new QB query.

The Query Builder can be accessed from the "Admin" menu when this option is turned on for the logged in user. The QB can be found under the "Reports" heading.

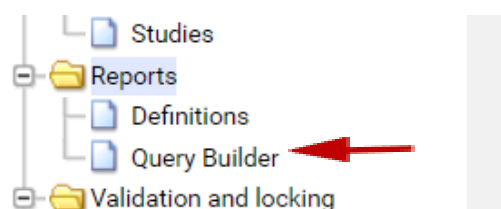


Figure 1: Query Builder option.

A list of defined queries appears under this menu item. To create a new query, start with the "new" button.

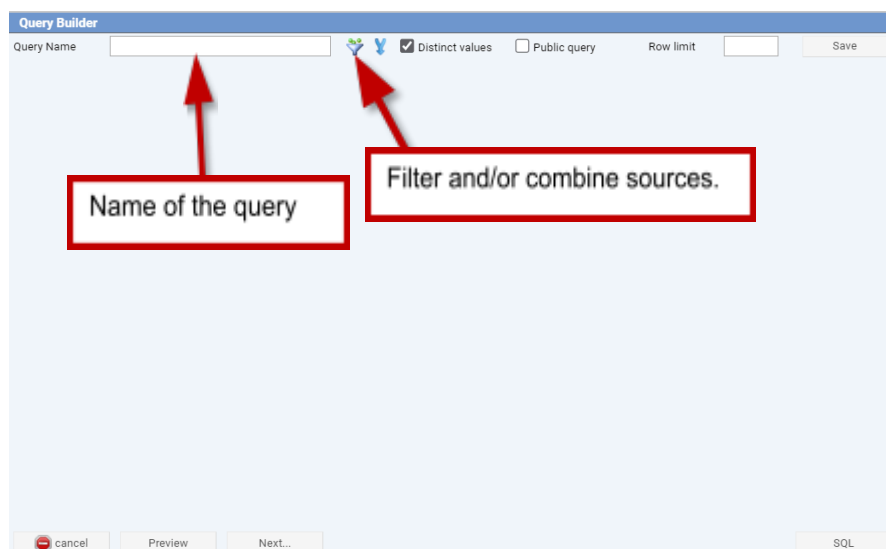


Figure 2: Name the query.

Step 1: select a source. This can be combined with other sources to expand the selection of the number of fields.

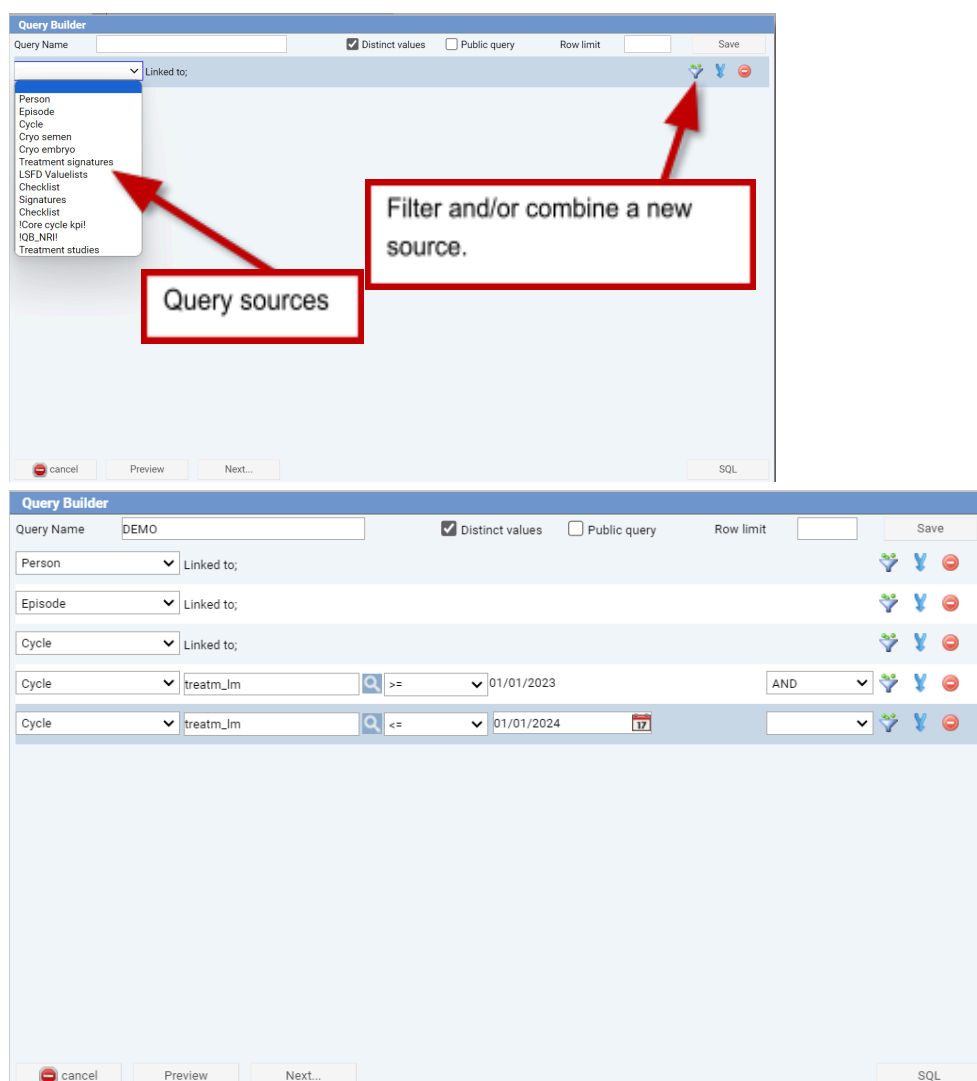
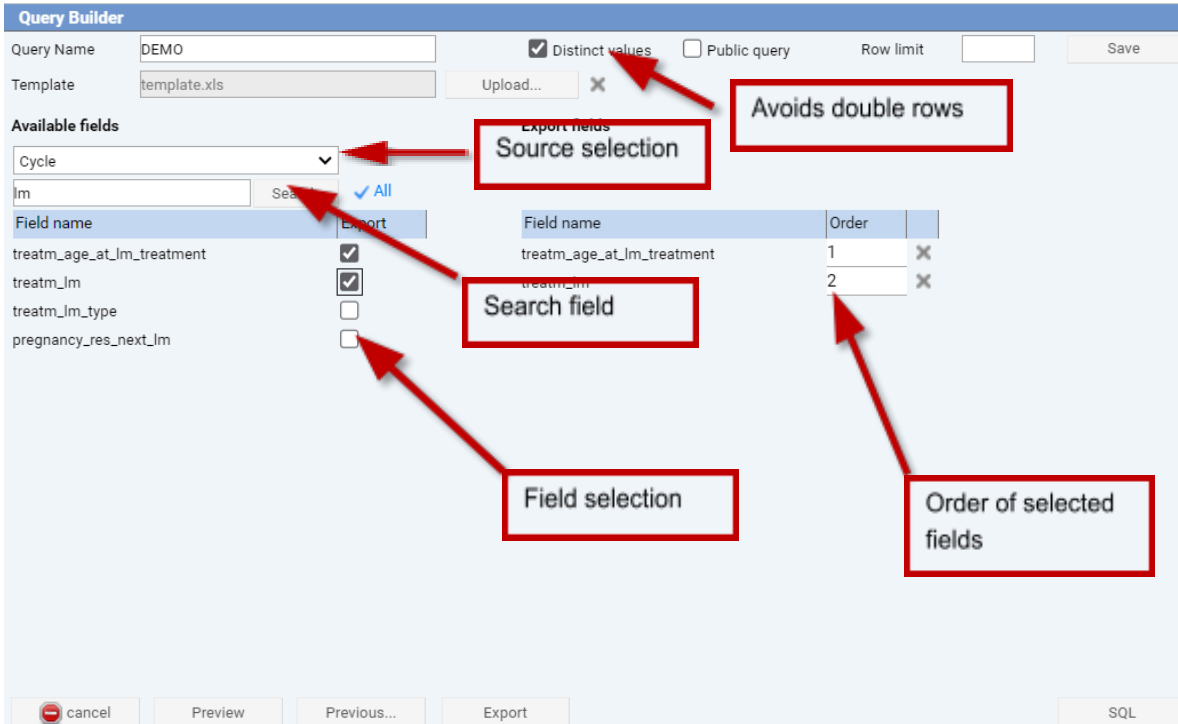


Figure 3: Choose a source.

Example in figure 3: persons linked to episodes and treatments, with a filter applied on treatments with LM date between 01/01/2023 and 01/01/2024.

2.2 Selecting Fields

Once the sources and filters have been determined, the "next" button can be used to open the field selection screen. Depending on the sources in this screen, fields from those sources can be selected here. This screen looks as follows:



Query Builder

Query Name: DEMO ☒ Distinct values ☐ Public query Row limit: Save

Template: template.xls Upload... X

Available fields

Cycle ☒ All

Field name	Selected
treatm_age_at_lm_treatment	<input checked="" type="checkbox"/>
treatm_lm	<input checked="" type="checkbox"/>
treatm_lm_type	<input type="checkbox"/>
pregnancy_res_next_lm	<input type="checkbox"/>

Export fields

Field name	Order	
treatm_age_at_lm_treatment	1	X
treatm_lm	2	X

cancel Preview Previous... Export SQL

Figure 4: Query Builder.

By selecting another source, the corresponding fields can be searched. A number of fields have so-called "aliases" present so that these can also be searched for. This is particularly necessary for fields that have a cryptic description.

2.3 Preview

By creating a "preview" the result can be viewed before exporting it into Excel. The "Row limit" option is useful here to limit the number of preview lines.

Preview (10 of 10) records	
treatm_lm	treatm_age_at_lm_treatment
2023-02-13 00:00:00.0	42
2023-01-06 00:00:00.0	57
2023-01-09 00:00:00.0	56
2023-02-09 00:00:00.0	53
2023-02-09 00:00:00.0	48
2023-01-05 00:00:00.0	54
2023-02-10 00:00:00.0	37
2023-01-02 00:00:00.0	41
2023-02-07 00:00:00.0	36
2023-10-28 00:00:00.0	40

Figure 5: preview view with a row limit of 10.

2.4 Export and save query

When a query is created, it can be saved and exported. By default, a query is only available to the logged-in user. By activating the "public query" check box, the export is available to all QB users.

An "Export" involves putting the list into an "Excel" file and downloading it. It is then possible to further analyze the data from the Excel file. Optionally, the modified and filled Excel can be linked back to the original query so that pivot tables, graphs and calculations are preserved.

3. Creating queries for custom reports

The second type is not based on the "standard" sources from the QB but gives the option to combine tables and fields completely "freely." A query can be used to retrieve data from the entire database. This is done with a so-called "SQL query. A SQL query in the eBase always starts with 'SELECT' and often ends with 'FROM'. After 'FROM' you can add additional commands to make your query more complex.

When you create a report in the eBase you have two choices: Excel export and Velocity report. The following explains the function of these different reporting options.

Excel export	It is used for complete analysis in Excel format. Example: the annual report.
Velocity report	These are (custom) reports that require specific formatting. Generally, this reporting capability is only used by eFertility.

Table 1: Overview of reporting options.

3.1 Components of an SQL query

The following table briefly describes the SQL components and provides an example of a raw query.

SQL component	Description
SELECT	Data retrieval is done with the 'SELECT' command. 'SELECT column name'.
FROM	Displays the tables containing the fields displayed in the SELECT component.
WHERE	If you want to select certain rows that must satisfy 1 or more conditions, or, conversely, must not satisfy 1 or more conditions, we use the 'WHERE' command.
GROUP BY	Periodically, you need information from tables that can be obtained by grouping values of columns or adding rows and columns together. The command 'GROUP BY' groups information together. The big difference with 'SELECT DISTINCT' is that you can do math with 'GROUP BY'.
ORDER BY	Data from a table can also be retrieved sorted. To sort, we use the command 'ORDER BY'. Sorting can be done in two ways: ascending and descending. In English, this is ascending (ASC) and descending (DESC).
HAVING	Suppose we want to know which function group has received a total of more than 5,000 messages we should use the HAVING command.
=	You give a fixed value (= 'value')
LIKE	With this command you guess the value. LIKE indicates it looks like.
INNER JOIN	This SQL command allows you to link two tables together, displaying the data from both tables only when the link is actually made.
LEFT JOIN	This SQL - command allows one to join two tables together with all records from the left side of the join being displayed regardless of whether there is a right side.
*	* means all fields in that table.
BETWEEN	Here you give a command that it is between 0 - 100.
NULL	The column may contain empty fields.
NOT NULL	The column must not contain empty fields.
AND / OR / NOT	If you want results that must meet multiple criteria you can use the 'AND', 'OR' and 'NOT' commands.

Table 2: view SQL components.

3.2 Sample query

The query below shows all data for a Treatment/Person/Puncture/Transfer:

```
SELECT care_person.*, care_treatment.*, care_puncture.*, care_inj_insem_trans.*
FROM care_treatment
INNER JOIN care_person ON care_person.care_person_id = care_treatment.care_person_id
LEFT JOIN care_puncture ON care_treatment.care_treatment_id = care_puncture.care_treatment_id
LEFT JOIN care_inj_insem_trans ON care_treatment.care_treatment_id =
care_inj_insem_trans.care_treatment_id
WHERE ISNULL(care_treatment.DELETED_CHK, 0) <> 1
```

This query can be written out in "colloquial language" as follows :

```
SELECT care_person.*, care_treatment.*, care_puncture.*, care_inj_insem_trans.*
Select all fields from the "care_person" (individuals), "care_treatment" (treatments),
"care_puncture" (punctures) and "care_inj_insem_trans" (inseminations and ETs) tables.
```

```
FROM care_treatment
From treatments.
```

```
INNER JOIN care_person ON care_person.care_person_id = care_treatment.care_person_id
linked to "care_person" via the fields "care_person_id" from the "care_person" and
"care_person_id" from "care_treatment". In doing so, only display treatments where a person is also
known (INNER JOIN).
```

```
LEFT JOIN care_puncture ON care_treatment.care_treatment_id = care_puncture.care_treatment_id
Then link the treatment to the puncture, regardless of whether it was performed.
```

```
LEFT JOIN care_inj_insem_trans ON care_treatment.care_treatment_id =
care_inj_insem_trans.care_treatment_id
As well as to any transfer/insemination.
```

```
WHERE ISNULL(care_treatment.DELETED_CHK, 0) <> 1
In doing so, omit all removed treatments.
```

In an Oracle environment, the function "ISNULL" does not exist. There, it is called "NVL." So there it would become "WHERE NVL(care_treatment.DELETED_CHK, 0) <> 1".