Base Guide 7.2

Appendix A
Common Database Tasks

Copyright

Contributors

To this edition

Pulkit Krishna
Jenna Sargent

To previous editions

Randolph Gamo
Jost Lange
Jochen Schiffers
Robert Großkopf
Hazel Russman
Jean Hollis Weber
Pulkit Krishna
Jenna Sargent

Feedback

Please direct any comments or suggestions about this document to the Documentation Team's mailing list: documentation@global.libreoffice.org.

Note

Everything you send to a mailing list, including your email address and any other personal information that is written in the message, is publicly archived and cannot be deleted.

Publication date and software version

Published December 2021. Based on LibreOffice 7.2 Community.
Other versions of LibreOffice may differ in appearance and functionality.

Contents

Barcodes

To be able to use the barcode print function, the font `ean13.ttf` must be installed. This font is freely available.

EAN13 barcodes can be created using `ean13.ttf` as follows:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Number	Upper case, A=0 B=1 etc.						*	Lower case, a=0 b=1 etc.						+

See also the query `Barcode_EAN13_ttf_command` in the example database Media_without_Macros.

Data types for the table editor

Integers

Type	***Option***	***HSQLDB***	***Range***	***Storage space***
Tiny Integer	TINYINT	TINYINT	2^8 = 256 | – 128 to + 127	1 Byte
Small Integer	SMALLINT	SMALLINT	2^{16} = 65536 | – 32768 to + 32767	2 Bytes
Integer	INTEGER	INTEGER | INT	2^{32} = 4294967296 | – 2147483648 to + 2147483647	4 Bytes
BigInt	BIGINT	BIGINT	2^{64}	8 Bytes

Floating-point numbers

Type	***Option***	***HSQLDB***	***Range***	***Storage space***
Decimal	DECIMAL	DECIMAL	Unlimited, up to 50 places in the GUI, fixed decimal point, perfect accuracy	variable
Number	NUMERIC	NUMERIC	Unlimited, up to 50 places in the GUI, fixed decimal point, perfect accuracy	variable
Float	FLOAT	(DOUBLE used instead)		
Real	REAL	REAL		

Double	DOUBLE	DOUBLE [PRECISION] | FLOAT	Adjustable, not exact, 15 decimal places maximum	8 Bytes

Text

Type	*Option*	*HSQLDB*	*Range*	*Storage space*
Text	VARCHAR	VARCHAR	Adjustable	variable
Text	VARCHAR_ IGNORECASE	VARCHAR_ IGNORECASE	Adjustable, range affects sorting	variable
Text (fix)	CHAR	CHAR | CHARACTER	Adjustable, rest of actual text replaced with spaces	fixed
Memo	LONGVARCHAR	LONGVARCHAR		variable

Time

Type	*Option*	*HSQLDB*	*Range*	*Storage space*
Date	DATE	DATE		4 Bytes
Time	TIME	TIME		4 Bytes
Date/Time	TIMESTAMP	TIMESTAMP | DATETIME	Adjustable (0.6 – 6 means with milliseconds)	8 Bytes

Other

Type	*Option*	*HSQLDB*	*Range*	*Storage space*
Yes/No	BOOLEAN	BOOLEAN | BIT		
Binaryfield (fix)	BINARY	BINARY	Like Integer	fixed
Binary field	VARBINARY	VARBINARY	Like Integer	variable
Image	LONGVARBINARY	LONGVARBINARY	Like Integer	variable, intended for larger images
OTHER	OTHER	OTHER | OBJECT		

In the table definitions, and when data types are changed in queries using the “convert” or “cast” functions, some data types expect information about the number of characters (a), the precision

(g, corresponding to the total number of characters) and the number of decimal places (d). The types are CHAR(a), VARCHAR(a), DOUBLE(g), NUMERIC(g, d), DECIMAL(g, d) and TIMESTAMP(g).

TIMESTAMP(g) can have only two values: ‘0’ and ‘6’. ‘0’ means that no seconds will be stored in the decimal part (tenths, hundredths...). The precision of timestamps can be given only *directly using SQL commands*. So if you are storing timings from some kind of sport, you must set TIMESTAMP(6) using **Tools > SQL** in advance.

Data types in StarBasic

Numbers

Type	*Corresponds to HSQLDB*	*Initial value*	*Remarks*	*Storage requirements*
Integer	SMALLINT	0	2^{16} = – 32768 bis + 32767	2 Bytes
Long	INTEGER	0	2^{32} = – 2147483648 bis + 2147483647	4 Bytes
Single		0.0	Decimal: .	4 Bytes
Double	DOUBLE	0.0	Decimal: .	8 Bytes
Currency	Resembles DECIMAL, NUMERIC	0.0000	4 fixed decimal places	8 Bytes

Others

Type	*Corresponds to HSQLDB*	*Initial value*	*Remarks*	*Storage requirements*
Boolean	BOOLEAN	False	1 = yes, everything else: no.	1 Byte
Date	TIMESTAMP	00:00:00	Date and time	8 Bytes
String	VARCHAR	Empty String	Up to 65536 characters	variable
Object	OTHER	Null		variable
Variant		Empty	Can accept any (other) data type	variable

There are great risks in data conversion, especially with numeric values. For example, primary keys in databases are most commonly of the type INTEGER. If these are read out by a macro, the variable in which they are stored must be of the type Long, as this corresponds in size to the INTEGER type in Base. The corresponding read instruction is `getLong`.

Built-in functions and stored procedures

The following functions are available in the built-in HSQLDB. Unfortunately one or two functions can only be used when **Run SQL command directly** is chosen. This will then prevent these queries from being edited.

Functions that work with the graphical user interface are marked [Works in the GUI]. Functions that work only in direct SQL commands are marked [Direct SQL – does not work in the GUI].

Numeric

As we are dealing here with floating point numbers, be sure to take care with the settings of the fields in queries. Mostly the display of decimal places is restricted, so that in some cases there may be unexpected results. For example, column 1 might show 0.00 but actually contain 0.001, and column 2, 1000. If column 3 is set to show Column 1 * Column 2, it would actually show 1.

ABS(d)	Returns the absolute value of a number. [Works in the GUI]
ACOS(d)	Returns the arccosine. [Works in the GUI]
ASIN(d)	Returns the arcsine. [Works in the GUI]
ATAN(d)	Returns the arctangent. [Works in the GUI]
ATAN2(a,b)	Returns the arctangent using coordinates, where a is the value of the x-axis, b the value of the y-axis. [Works in the GUI]
BITAND(a,b)	Both the binary form of a and the binary form of b must have 1 at the same position to yield 1 in the result. BITAND(3,5) yields 1; 0011 AND 0101 = 0001 [Works in the GUI]
BITOR(a,b)	Either the binary form of a or the binary form of b must have 1 at the same position to yield 1 in the result. BITOR(3,5) yields 7; 0011 OR 0101 = 0111 [Works in the GUI]
CEILING(d)	Returns the smallest whole number that is not smaller than d. [Works in the GUI]
COS(d)	Returns the cosine. [Works in the GUI]
COT(d)	Returns the cotangent. [Works in the GUI]
DEGREES(d)	Converts radians to degrees. [Works in the GUI]
EXP(d)	Returns e^d (e: (2.718...)). [Works in the GUI]
FLOOR(d)	Returns the largest whole number that is not greater than d. [Works in the GUI]
LOG(d)	Returns the natural logarithm to base e. [Works in the GUI]

LOG10(d)	Returns the logarithm to base 10. [Works in the GUI]
MOD(a,b)	Returns the remainder as a whole number, in the division of 2 whole numbers. MOD(11,3) returns 2, because 3*3+2=11 [Works in the GUI]
PI()	Returns π (3.1415...). [Works in the GUI]
POWER(a,b)	a^b , POWER(2,3) = 8, since $2^3 = 8$ [Works in the GUI]
RADIANS(d)	Converts degrees to radians. [Works in the GUI]
RAND()	Returns a random number greater than or equal to 0.0 and less than 1.0. [Works in the GUI]
ROUND(a,b)	Rounds a to b decimal places. [Works in the GUI]
ROUNDMAGIC(d)	Solves rounding problems that arise from using floating point numbers. 3.11-3.1-0.01 is not exactly 0, but is shown as 0 in the GUI. ROUNDMAGIC makes it an actual zero value. [Works in the GUI]
SIGN(d)	Returns –1 if d is less than 0, 0 if d is equal to 0, and 1 if d is greater than 0. [Works in the GUI]
SIN(A)	Returns the sine of an angle in radians. [Works in the GUI]
SQRT(d)	Returns the square root. [Works in the GUI]
TAN(A)	Returns the tangent of an angle in radians. [Works in the GUI]
TRUNCATE(a,b)	Truncates a to b decimal places. TRUNCATE(2.37456,2) = 2.37 [Works in the GUI]
Text	
ASCII(s)	Returns the ASCII code of the first letter of the string. [Works in the GUI]
BIT_LENGTH(str)	Returns the length of the text string str in bits. [Works in the GUI]
CHAR(c)	Returns the letter corresponding to the ASCII code c. [Works in the GUI]

CHAR_LENGTH(str)	Returns the length of the string str in characters. [Works in the GUI]
CONCAT(str1,str2)	Concatenates str1 and str2. [Works in the GUI]
'str1'||'str2'||'str3' or 'str1'+'str2'+'str3'	Concatenates str1, str2, and str3. A simpler alternative to CONCAT. [Works in the GUI]
DIFFERENCE(s1,s2)	Returns the sound difference between s1 and s2. Only a whole number is output. 0 means they sound the same. For example, 'for' and 'four' yield 0, ‘king’ and ‘wing’ yield 1, ‘see’ and ‘sea’ yield 0. [Works in the GUI]
HEXTORAW(s1)	Translates hexadecimal code to other characters. [Works in the GUI]
INSERT(s,start,len,s2)	Returns a text string, with part of the text replaced. Beginning with start, a length len is cut out of the text s and replaced by the text s2. INSERT(“Bundesbahn”, 3, 4, mmel) converts Bundesbahn into Bummelbahn, where the length of the inserted text can be greater than that of the deleted text without causing any problems. So INSERT(“Bundesbahn”, 3, 5, s und B) yields 'Bus und Bahn'. [Works in the GUI]
LCASE(s)	Converts a string to lower case. [Works in the GUI]
LEFT(s,count)	Returns the number of characters specified by count from the beginning of the string s. [Works in the GUI]
LENGTH(s)	Returns the length of string s in characters. [Works in the GUI]
LOCATE(search,s,[start])	Returns the first match for the term search in the string s. The match is given as an offset number: (1=left, 0=not found). Setting a starting point within the text string is optional. [Works in the GUI]
LTRIM(s)	Removes leading spaces and non-printing characters from the beginning of a text string. [Works in the GUI]
OCTET_LENGTH(str)	Returns the length of a text string in bytes. This corresponds to twice the length of the string. [Works in the GUI]
RAWTOHEX(s1)	Converts to hexadecimals, reverse of HEXTORAW(). [Works in the GUI]
REPEAT(s,count)	Repeats the text string s count times. [Works in the GUI]
REPLACE(s,replace,s2)	Replaces all existing occurrences of replace in the text string s by the string s2. [Works in the GUI]

RIGHT(s,count)	Opposite of LEFT; returns the last count characters at the end of a text string. [Works in the GUI]
RTRIM(s)	Removes all spaces and non-printing characters from the end of a text string. [Works in the GUI]
SOUNDEX(s)	Returns a 4-character code, corresponding to the sound of s. Matches the function DIFFERENCE(). [Works in the GUI]
SPACE(count)	Returns count spaces. [Works in the GUI]
SUBSTR(s,start[,len])	Abbreviation for SUBSTRING. [Works in the GUI]
SUBSTRING(s,start[,len])	Returns the text s from the start position (1=left). If length len is left out, the whole string is returned. [Works in the GUI]
UCASE(s)	Converts a string to upper case. [Works in the GUI]
LOWER(s)	As LCASE(s). [Works in the GUI]
UPPER(s)	As UCASE(s). [Works in the GUI]
Date/Time	
CURDATE()	Returns the current date. [Works in the GUI]
CURTIME()	Returns the current time. [Works in the GUI]
DATEDIFF(string, datetime1, datetime2)	Date difference between two dates - compares date/time values. The entry in string determines the units in which the difference is returned: ms=millisecond, ss=second, mi=minute, hh=hour, dd=day, mm=month, yy = year. Both the long and the short forms can be used for string. [Works in the GUI]
DAY(date)	Returns the day of the month (1-31). [Works in the GUI]
DAYNAME(date)	Returns the English name of the day. [Works in the GUI]
DAYOFMONTH(date)	Returns the day of the month (1-31). Synonym for DAY(). [Works in the GUI]
DAYOFWEEK(date)	Returns the weekday as a number (1 represents Sunday). [Works in the GUI]

DAYOFYEAR(date)	Returns the day of the year (1-366). [Works in the GUI]
HOUR(time)	Returns the hour (0-23). [Works in the GUI]
MINUTE(time)	Returns the minute (0-59). [Works in the GUI]
MONTH(date)	Returns the month (1-12). [Works in the GUI]
MONTHNAME(date)	Returns the English name of the month. [Works in the GUI]
NOW()	Returns the current date and the current time together as a timestamp. Alternatively CURRENT_TIMESTAMP can be used. [Works in the GUI]
QUARTER(date)	Returns the quarter of the year (1-4). [Works in the GUI]
SECOND(time)	Returns the seconds part of the time (0-59). [Works in the GUI]
WEEK(date)	Returns the week of the year (1-53). [Works in the GUI]
YEAR(date)	Returns the year part of a date entry. [Works in the GUI]
CURRENT_DATE	Synonym for CURDATE(), SQL-Standard. [Works in the GUI]
CURRENT_TIME	Synonym for CURTIME(), SQL-Standard. [Works in the GUI]
CURRENT_TIMESTAMP	Synonym for NOW(), SQL-Standard. [Works in the GUI]

Database connection

Except for IDENTITY(), which has no meaning in Base, all of these can be carried out using **Direct SQL Command**.

DATABASE()	Returns the name of the database to which this connection belongs. [Works in the GUI]
USER()	Returns the username of this connection. [Direct SQL – does not work with the GUI]
CURRENT_USER	SQL standard function, synonym for USER(). [Works in the GUI]
IDENTITY()	Returns the last value for an autovalue field, which was created in the current connection. This is used in macro coding to transfer a primary key in one table to become a foreign key for another table. [Works in the GUI]

System	
IFNULL(exp,value)	If exp is NULL, value is returned, otherwise exp is returned. Alternatively as an extension COALESCE() can be used. Exp and value must have the same data type. [Works in the GUI]
CASEWHEN(exp,v1,v2)	If exp is true, v1 is returned, otherwise v2. Alternatively CASE WHEN can be used. CASE WHEN works better with the GUI. [Works in the GUI]
CONVERT(term,type)	Converts term into another data type. [Works in the GUI]
CAST(term AS type)	Synonym for CONVERT(). [Works in the GUI]
COALESCE(expr1,expr2, expr3,...)	If expr1 is not NULL, returns expr1, otherwise expr2 is checked, then expr3 and so on. [Works in the GUI]
NULLIF(v1,v2)	If v1 is equal to v2, NULL is returned, otherwise v1 is returned. [Works in the GUI]
CASE v1 WHEN v2 THEN v3 [ELSE v4] END	If v1 is equal to v2, v3 is returned. Otherwise v4 is returned or NULL, if there is no ELSE condition. [Direct SQL – does not work with the GUI]
CASE WHEN expr1 THEN v1[WHEN expr2 THEN v2] [ELSE v4] END	If expr1 is true, v1 is returned [optionally further conditions can be set]. Otherwise v4 is returned or NULL if there is no ELSE condition. [Works in the GUI]
EXTRACT ({YEAR | MONTH | DAY | HOUR | MINUTE | SECOND} FROM <date or time>)	Can replace many of the date and time functions. Returns the year, month, day, etc. from a date or date/time value. [Works in the GUI]
POSITION(<string expression> IN <string expression>)	If the first string is contained in the second one, the offset of the first string is given, otherwise 0 is returned. [Works in the GUI]
SUBSTRING(<string expression> FROM <numeric expression> [FOR <numeric expression>])	Yields part of a text string from the position specified in FROM, optionally up to the length given in FOR. [Works in the GUI]
TRIM([{LEADING | TRAILING | BOTH}] FROM <string expression>)	Non-printing special characters and spaces are removed. [Works in the GUI]

Control characters for use in queries

Fields can be linked together in queries. Two fields in

```
SELECT "First name", "Surname" FROM "Table"
```

become one field by using:

```
SELECT "First name"||' '||"Surname" FROM "Table"
```

Here an additional space is inserted. It could be any character; as long as it is enclosed in `''`, it will be interpreted as text. Sometimes, however, it is necessary to insert non-printing characters such as new lines, for example in preparing reports. Here is a short list of control characters. More information on these is available at https://en.wikipedia.org/wiki/Control_character.

CHAR(9)	Horizontal Tab	
CHAR(10)	Line feed	In mail merge letters and Report Builder, creates a line break (Linux, Unix, Mac).
CHAR(13)	Carriage return	Line break when combined with Carriage return in Windows `CHAR(13)||CHAR(10)`. Can also be used in Linux and Mac, hence the universal variant.

Some uno commands for use with a button

A button can have various uno commands directly bound to it. For this purpose you need to choose **Properties: Button > Action > Open document/web page** and then for example the **URL > .uno:RecSearch** to open the search function. Often you will need to choose **Take Focus on Click > No** if the action accesses another control directly in a way that requires it to be in focus, for example **.uno:Paste**, which can insert the contents of the clipboard.

The following list contains only a few commands. All the commands from the navigation toolbar are already usable in the button, but they can also be created using uno commands. Many commands can be discovered by using the macro recorder, which often uses a dispatcher to access them.

Uno-Command	***Used for ...***
.uno:RecSearch	Opens the search function in a form.
.uno:Paste	Paste from clipboard. Only works for **Take Focus on Click > No**
.uno:Copy	Copies the selected content onto the clipboard. Only works for **Take Focus on Click > No**
.uno:Print	Opens the print dialog for the form.
.uno:PrintDefault	Prints with the default printer without showing a dialog.

Information tables for HSQLDB

Inside a database, information on all table properties and their connections to one another are stored in the *INFORMATION_SCHEMA* area. This information allows Base macros to be created that require very few arguments for their procedures. An application is given in the example database in the Maintenance module—the Table_purge procedure for the control of dialogs.

In a query, individual pieces of information and all the fields that belong can be provided in the following way:

```
SELECT * FROM "INFORMATION_SCHEMA"."SYSTEM_ALIASES"
```

In contrast to a normal table, it is necessary here to use *INFORMATION_SCHEMA* as a prefix to the appropriate name from the following list:

```
SYSTEM_ALIASES
SYSTEM_ALLTYPEINFO
SYSTEM_BESTROWIDENTIFIER
SYSTEM_CACHEINFO
SYSTEM_CATALOGS
SYSTEM_CHECK_COLUMN_USAGE
SYSTEM_CHECK_CONSTRAINTS
SYSTEM_CHECK_ROUTINE_USAGE
SYSTEM_CHECK_TABLE_USAGE
SYSTEM_CLASSPRIVILEGES
SYSTEM_COLUMNPRIVILEGES
SYSTEM_COLUMNS
SYSTEM_CROSSREFERENCE
SYSTEM_INDEXINFO
SYSTEM_PRIMARYKEYS
SYSTEM_PROCEDURECOLUMNS
SYSTEM_PROCEDURES
SYSTEM_PROPERTIES
SYSTEM_SCHEMAS
SYSTEM_SEQUENCES
SYSTEM_SESSIONINFO
SYSTEM_SESSIONS
SYSTEM_SUPERTABLES
SYSTEM_SUPERTYPES
SYSTEM_TABLEPRIVILEGES
SYSTEM_TABLES
SYSTEM_TABLETYPES
SYSTEM_TABLE_CONSTRAINTS
SYSTEM_TEXTTABLES
SYSTEM_TRIGGERCOLUMNS
SYSTEM_TRIGGERS
SYSTEM_TYPEINFO
SYSTEM_UDTATTRIBUTES
SYSTEM_UDTS
SYSTEM_USAGE_PRIVILEGES
SYSTEM_USERS
SYSTEM_VERSIONCOLUMNS
SYSTEM_VIEWS
SYSTEM_VIEW_COLUMN_USAGE
SYSTEM_VIEW_ROUTINE_USAGE
SYSTEM_VIEW_TABLE_USAGE
```

The following query gives a complete overview of all tables in the database with field types, primary keys and foreign keys:

```
SELECT
"A"."TABLE_NAME",
"A"."COLUMN_NAME",
"A"."TYPE_NAME",
"A"."NULLABLE",
```

```
"B"."KEY_SEQ" AS "PRIMARYKEY",
"C"."PKTABLE_NAME" || '.' || "C"."PKCOLUMN_NAME" AS "FOREIGNKEY FOR"
FROM "INFORMATION_SCHEMA"."SYSTEM_COLUMNS" AS "A"
   LEFT JOIN "INFORMATION_SCHEMA"."SYSTEM_PRIMARYKEYS" AS "B"
   ON ( "B"."TABLE_NAME" = "A"."TABLE_NAME" AND "B"."COLUMN_NAME" =
   "A"."COLUMN_NAME" )
   LEFT JOIN "INFORMATION_SCHEMA"."SYSTEM_CROSSREFERENCE" AS "C"
   ON ( "C"."FKTABLE_NAME" = "A"."TABLE_NAME" AND
   "C"."FKCOLUMN_NAME"    = "A"."COLUMN_NAME" )
WHERE "A"."TABLE_SCHEM" = 'PUBLIC'
```

Database repair for *.odb files

Regular data backups should be standard practice when using a PC. Backup copies are the simplest way to return to an even halfway current state for your data. However, in practice this is often lacking.

Forms, queries, and reports can always be copied using the clipboard into a new database, providing that a previous version of the database has been saved. But if, for any reason, the current database can no longer be opened, the main problem becomes access to the data.

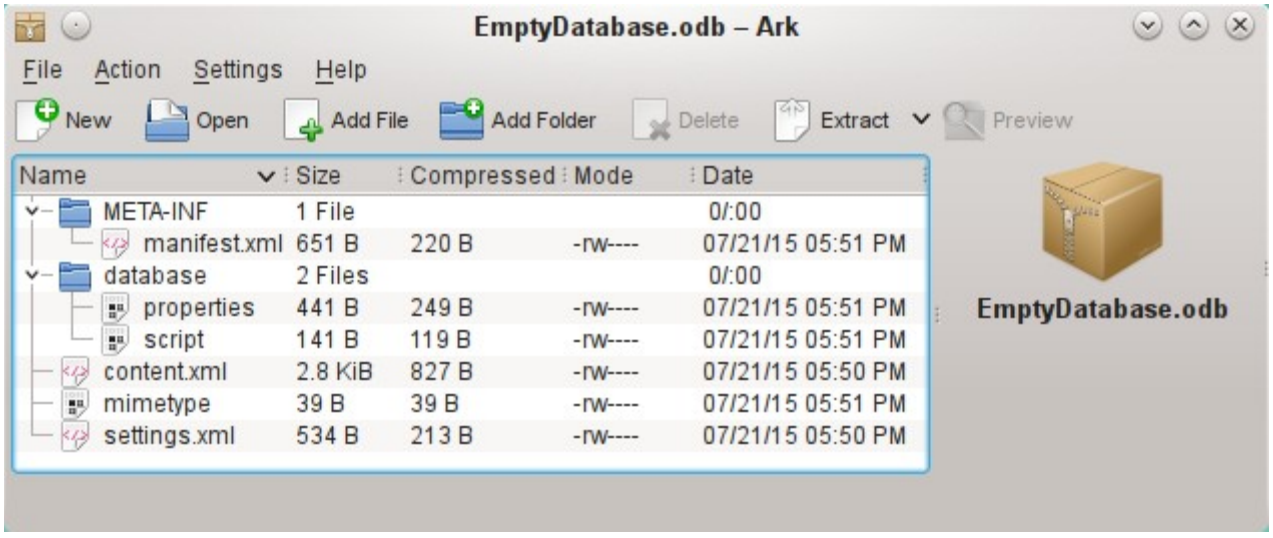
In the case of sudden PC crashes, it can happen that open databases (internal HSQLDB databases) can no longer be opened in LibreOffice. Instead, when you attempt to open the database, you are asked for a filter corresponding to the format.

The problem here is that part of the data in an open database is contained in working memory and is only temporarily copied to intermediate storage. Only when the file is closed is the whole database written back into the file and repacked.

Recovery of the database archive file

To get access again to your data, you may find the following procedure helpful:

1) Create a copy of your database for the steps that follow.
2) Try to open the copy with an archiving program. In the case of *.odb files, we are dealing with a compressed format, a Zip archive. If the file cannot be opened directly, try renaming it from *.odb to *.zip. If that does not open it, your database is past saving.
3) The following folders will always be seen after opening a database file in an archiving program:



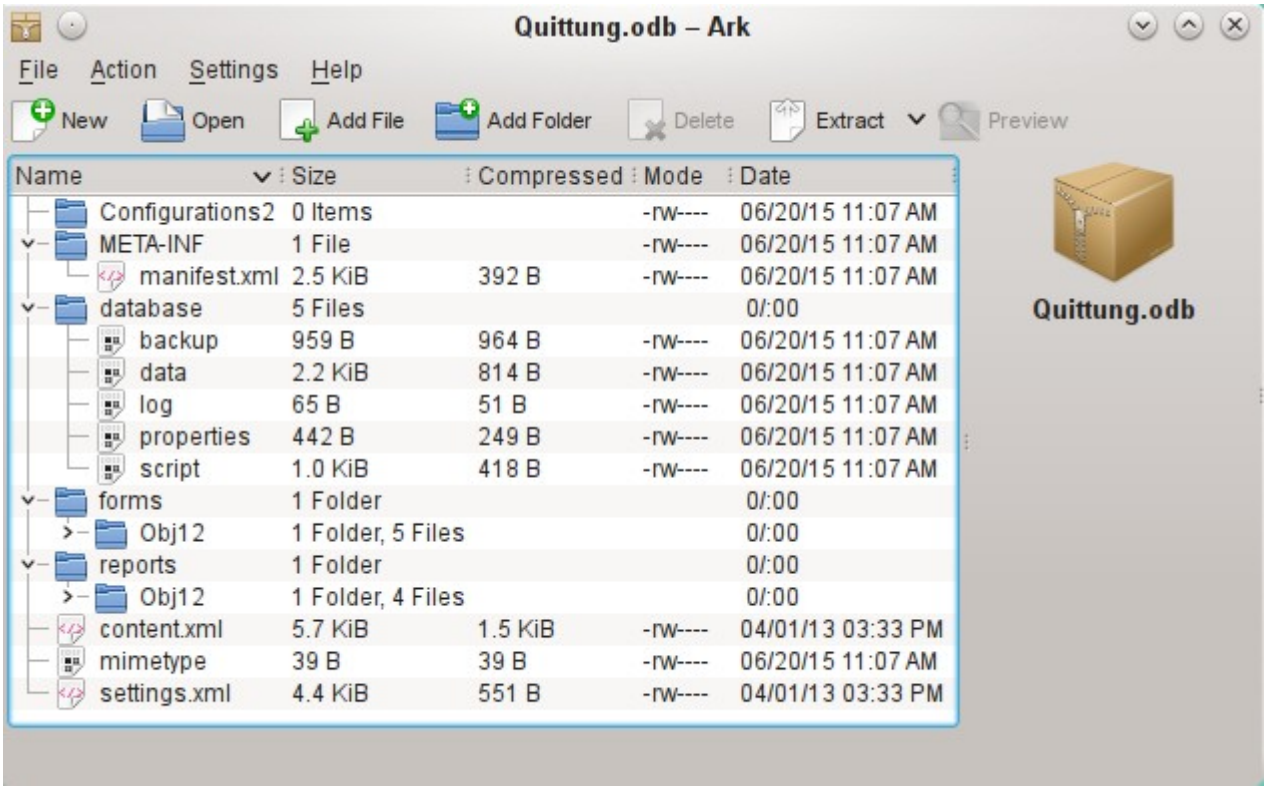
4) The database file must be decompressed. The most important information, as far as the data is concerned, is in the subfolder database in the files `data` and `script`.
5) It may be necessary to look at the `script` file and test it for contradictions. This step can, however, be left for the testing stage. The `script` file contains above all the description of the table structure.
6) Create a new empty database file and open this file with the archiving program.
7) Replace the files `data` and `script` in the new database file with the files unpacked in step 4.
8) Close the archiving program. If it was necessary to rename the file to *.zip before opening it in the archiving program (this depends on your operating system), now rename it again to *.odb.
9) Open the database file in LibreOffice. You should be able to access your tables again.
10) How much of your queries, forms, and reports can be recovered in a similar way must be the subject of further testing.

See also: http://forum.openoffice.org/en/forum/viewtopic.php?f=83&t=17125

Further information on database archive files

In practice, a database archive file contains not only the basic folder for the database, and the folder META-INF which is specified for the OpenDocument format, but also additional folders for storing forms and reports. A description of the basic structure of the OpenDocument format can be found at https://en.wikipedia.org/wiki/OpenDocument_technical_specification.

The following view shows a database containing tables, a form and a report. It is not apparent that the database also contains a query. Queries are not stored in separate folders but in the content.xml file. The information necessary to run a query is a simple piece of SQL code.



Database file which contains stored information for a form and a report in addition to the database.

Here is an overview of one of the database archive files.

mimetype

```
application/vnd.oasis.opendocument.base
```

eine

This little text file contains only the notice that this archive file is a database file in OpenDocument format.

content.xml for a database without content

```
<?xml version="1.0" encoding="UTF-8"?>
<office:document-content
xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
xmlns:style="urn:oasis:names:tc:opendocument:xmlns:style:1.0"
xmlns:text="urn:oasis:names:tc:opendocument:xmlns:text:1.0"
xmlns:table="urn:oasis:names:tc:opendocument:xmlns:table:1.0"
xmlns:draw="urn:oasis:names:tc:opendocument:xmlns:drawing:1.0"
xmlns:fo="urn:oasis:names:tc:opendocument:xmlns:xsl-fo-compatible:1.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:meta="urn:oasis:names:tc:opendocument:xmlns:meta:1.0"
xmlns:number="urn:oasis:names:tc:opendocument:xmlns:datastyle:1.0"
xmlns:svg="urn:oasis:names:tc:opendocument:xmlns:svg-compatible:1.0"
xmlns:chart="urn:oasis:names:tc:opendocument:xmlns:chart:1.0"
xmlns:dr3d="urn:oasis:names:tc:opendocument:xmlns:dr3d:1.0"
xmlns:math="http://www.w3.org/1998/Math/MathML"
xmlns:form="urn:oasis:names:tc:opendocument:xmlns:form:1.0"
xmlns:script="urn:oasis:names:tc:opendocument:xmlns:script:1.0"
xmlns:ooo="http://openoffice.org/2004/office"
xmlns:ooow="http://openoffice.org/2004/writer"
xmlns:oooc="http://openoffice.org/2004/calc"
xmlns:dom="http://www.w3.org/2001/xml-events"
xmlns:db="urn:oasis:names:tc:opendocument:xmlns:database:1.0"
xmlns:xforms="http://www.w3.org/2002/xforms"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:rpt="http://openoffice.org/2005/report"
xmlns:of="urn:oasis:names:tc:opendocument:xmlns:of:1.2"
xmlns:xhtml="http://www.w3.org/1999/xhtml"
xmlns:grddl="http://www.w3.org/2003/g/data-view#"
xmlns:tableooo="http://openoffice.org/2009/table"
xmlns:drawooo="http://openoffice.org/2010/draw"
xmlns:calcext="urn:org:documentfoundation:names:experimental:calc:xmlns:calcext:1.0"
xmlns:field="urn:openoffice:names:experimental:ooo-ms-interop:xmlns:field:1.0"
xmlns:formx="urn:openoffice:names:experimental:ooxml-odf-interop:xmlns:form:1.0"
xmlns:css3t="http://www.w3.org/TR/css3-text/"
office:version="1.2">
   <office:scripts/>
   <office:font-face-decls/>
   <office:automatic-styles/>
   <office:body>
      <office:database>
         <db:data-source>
            <db:connection-data>
               <db:connection-resource xlink:href="sdbc:embedded:hsqldb"/>
               <db:login db:is-password-required="false"/>
            </db:connection-data>
            <db:driver-settings
               db:system-driver-settings=""
               db:base-dn=""
               db:parameter-name-substitution="false"/>
            <db:application-connection-settings
               db:is-table-name-length-limited="false"
               db:append-table-alias-name="false"
               db:max-row-count="100">
               <db:table-filter>
```

```
                  <db:table-include-filter>
                     <db:table-filter-pattern>%</db:table-filter-pattern>
                  </db:table-include-filter>
               </db:table-filter>
            </db:application-connection-settings>
         </db:data-source>
      </office:database>
   </office:body>
</office:document-content>
```

It begins with the XML version and the character set used. Everything that follows is actually a single unwrapped line. The view prepared above should make things clearer. Elements that belong together are bracketed by tags.

The initial definitions beginning with `xmlns:` (XML namespace) give the namespaces that can be accessed from inside the file. Then somewhat more concrete terms are considered. Here it becomes clear that we are dealing with an internal HSQLDB database, and that a password is not required for access.

content.xml for a database with contents

The following content is only an excerpt from the content.xml file, to clarify its structure.

```
<office:scripts/>
<office:font-face-decls>
   <style:font-face style:name="F" svg:font-family=""/>
</office:font-face-decls>
<office:automatic-styles>
   <style:style
      style:name="co1"
      style:family="table-column"
      style:data-style-name="N0"/>
   <style:style
      style:name="co2"
      style:family="table-column"
      style:data-style-name="N107"/>
   <style:style style:name="ce1" style:family="table-cell">
      <style:paragraph-properties fo:text-align="start"/>
   </style:style>
   <number:number-style style:name="N0" number:language="de" number:country="DE">
      <number:number number:min-integer-digits="1"/>
   </number:number-style>
   <number:currency-style
      style:name="N107P0"
      style:volatile="true"
      number:language="de"
      number:country="DE">
      <number:number
         number:decimal-places="2"
         number:min-integer-digits="1"
         number:grouping="true"/>
      <number:text> </number:text>
      <number:currency-symbol
         number:language="de"
         number:country="DE">€
      </number:currency-symbol>
   </number:currency-style>
```

Here a field is defined as a currency field. The number of decimal places is given, the separation between the numbers and the currency symbol, and the currency symbol itself.

```
   <number:currency-style
      style:name="N107"
      number:language="de"
      number:country="DE">
      <style:text-properties fo:color="#ff0000"/>
      <number:text>-</number:text>
      <number:number
         number:decimal-places="2"
```

```
            number:min-integer-digits="1"
            number:grouping="true"/>
         <number:text> </number:text>
         <number:currency-symbol
            number:language="de"
            number:country="DE">€
         </number:currency-symbol>
         <style:map style:condition="value()&gt;=0" style:apply-style-name="N107P0"/>
      </number:currency-style>
```

The second extract states that up to a particular value, currency should appear in red (“ff0000”).

```
   </office:automatic-styles>
   <office:body>
      <office:database>
         <db:data-source>
```

This entry from the above content.xml file, with all its subentries, corresponds to an empty database archive file.

```
         </db:data-source>
         <db:forms>
            <db:component
               db:name="Receipts"
               xlink:href="forms/Obj12"
               db:as-template="false"/>
         </db:forms>
```

The database archive file contains a subsection in which details of a form are stored. The form is designated in the user interface as *Receipts*.

```
         <db:reports>
            <db:component
               db: name="Receipts"
               xlink:href="reports/Obj12"
               db:as-template="false"/>
         </db:reports>
```

The database archive file also contains a subsection in which details of a report are stored. The report is also designated in the user interface as *Receipts*.

```
         <db:queries>
            <db:query
               db:name="Sales_calc"
               db:command="SELECT &quot;a&quot;.*, ( SELECT &quot;Price&quot; *
                  &quot;a&quot;.&quot;Total&quot; FROM &quot;Stock&quot; WHERE
                  &quot;ID&quot; = &quot;a&quot;.&quot;Stock_ID&quot; ) AS
                  &quot;Total*Price&quot; FROM &quot;Sales&quot; AS &quot;a&quot;"/>
         </db:queries>
```

All the queries are stored directly in content.xml. " stands for double quotes. The query above in this example is actually quite complicated with many correlated subqueries. It is reproduced here in an abbreviated form.

```
         <db:table-representations>
            <db:table-representation db:name="Receipts"/>
            <db:table-representation db:name="Sales"/>
            <db:table-representation db:name="Stock">
               <db:columns>
                  <db:column
                     db:name="ID"
                     db:style-name="co1"
                     db:default-cell-style-name="ce1"/>
                  <db:column
                     db:name="MWSt"
                     db:style-name="co1"
                     db:gefault-cell-style-name="ce1"/>
                  <db:column
                     db:name="Price"
                     db:style-name="co2"
                     db:default-cell-style-name="ce1"/>
```

```
                <db:column
                   db:name="Stock"
                   db:style-name="co1"
                   db:default-cell-style-name="ce1"/>
             </db:columns>
          </db:table-representation>
       </db:table-representations>
```

This shows how various tables are to be displayed. Here the display properties of particular columns are stored: in this example, settings for the Stock table with its fields – ID, MWSt and so on – are stored. Apparently something has been directly entered here, changing the columns of the table a bit.

```
    </office:database>
</office:body>
```

Basically, content.xml stores directly the contents of queries and information about the visual appearance of tables. In addition there is a definition of the database connection. Finally comes information about forms and reports.

settings.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<office:document-settings
xmlns:office="urn:oasis:names:tc:opendocument:xmlns:office:1.0"
xmlns:table="urn:oasis:names:tc:opendocument:xmlns:table:1.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:number="urn:oasis:names:tc:opendocument:xmlns:datastyle:1.0"
xmlns:svg="http://www.w3.org/2000/svg"
xmlns:config="urn:oasis:names:tc:opendocument:xmlns:config:1.0"
xmlns:ooo="http://openoffice.org/2004/office"
xmlns:db="urn:oasis:names:tc:opendocument:xmlns:database:1.0"
office:version="1.2"/>
```

For a database without further content, only basic definitions are stored here. With content, various settings are also stored. After the start of the above definition, the following settings from the example database are stored.

```
<office:settings>
   <config:config-item-set config:name="ooo:view-settings">
      <config:config-item-set config:name="Queries">
         <config:config-item-set config:name="Calculate_sales">
            <config:config-item-set config:name="Tables">
               <config:config-item-set config:name="Table1">
                  <config:config-item config:name="WindowName"
                     config:type="string">Verkauf</config:config-item>
                  <config:config-item config:name="WindowLeft"
                     config:type="int">153</config:config-item>
                  <config:config-item config:name="ShowAll"
                     config:type="boolean">true</config:config-item>
                  <config:config-item config:name="WindowTop"
                     config:type="int">17</config:config-item>
                  <config:config-item config:name="WindowWidth"
                     config:type="int">120</config:config-item>
                  <config:config-item config:name="WindowHeight"
                     config:type="int">120</config:config-item>
                  <config:config-item config:name="ComposedName"
                     config:type="string">Verkauf</config:config-item>
                  <config:config-item config:name="TableName"
                     config:type="string">Verkauf</config:config-item>
               </config:config-item-set>
            </config:config-item-set>
            <config:config-item config:name="SplitterPosition"
               config:type="int">105</config:config-item>
            <config:config-item config:name="VisibleRows"
               config:type="int">1024</config:config-item>
         </config:config-item-set>
      </config:config-item-set>
   </config:config-item-set>
   <config:config-item-set config:name="ooo:configuration-settings">
```

```
        <config:config-item-set config:name="layout-settings">
           <config:config-item-set config:name="Tables">
              <config:config-item-set config:name="Table1">
                 <config:config-item config:name="WindowName"
                    config:type="string">Verkauf</config:config-item>
                 <config:config-item config:name="WindowLeft"
                    config:type="int">186</config:config-item>
                 <config:config-item config:name="ShowAll"
                    config:type="boolean">false</config:config-item>
                 <config:config-item config:name="WindowTop"
                    config:type="int">17</config:config-item>
                 <config:config-item config:name="WindowWidth"
                    config:type="int">120</config:config-item>
                 <config:config-item config:name="WindowHeight"
                    config:type="int">120</config:config-item>
                 <config:config-item config:name="ComposedName"
                    config:type="string">Verkauf</config:config-item>
                 <config:config-item config:name="TableName"
                    config:type="string">Sales</config:config-item>
              </config:config-item-set>
              <config:config-item-set config:name="Table2">
                 ... (identical config:type-Points as "Table1"
                 <config:config-item config:name="TableName"
                    config:type="string">Ware</config:config-item>
              </config:config-item-set>
              <config:config-item-set config:name="Table3">
                 ... (identical config:type-Points as "Table1"
                 <config:config-item config:name="TableName"
                    config:type="string">Receipts</config:config-item>
              </config:config-item-set>
           </config:config-item-set>
        </config:config-item-set>
     </config:config-item-set>
  </office:settings>
```

The whole overview relates to different views of windows for the query Calculate_sales and the tables Sales, Stock, and Receipts. The last two are shown here in an abbreviated form. If these settings were absent in a defective *.odb file, it would not matter. They would be recreated when the corresponding window was next opened.

META-INF/manifest.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest:manifest
   xmlns:manifest="urn:oasis:names:tc:opendocument:xmlns:manifest:1.0">
   <manifest:file-entry
      manifest:full-path="/"
      manifest:media-type="application/vnd.oasis.opendocument.base"/>
   <manifest:file-entry
      manifest:full-path="database/script"
      manifest:media-type=""/>
   <manifest:file-entry
      manifest:full-path="database/properties"
      manifest:media-type=""/>
   <manifest:file-entry
      manifest:full-path="settings.xml"
      manifest:media-type="text/xml"/>
   <manifest:file-entry
      manifest:full-path="content.xml"
      manifest:media-type="text/xml"/>
</manifest:manifest>
```

This file in the META-INF folder gives the contents folder for the whole database archive. As this file deals with an empty database, there are only five file entries. A database archive that contains forms and reports will have a much more complicated META-INF file.

database/properties

```
#HSQL Database Engine 1.8.0.10
#Sun Jul 14 18:02:08 CEST 2013
```

```
hsqldb.script_format=0
runtime.gc_interval=0
sql.enforce_strict_size=true
hsqldb.cache_size_scale=8
readonly=false
hsqldb.nio_data_file=false
hsqldb.cache_scale=13
version=1.8.0
hsqldb.default_table_type=cached
hsqldb.cache_file_scale=1
hsqldb.lock_file=true
hsqldb.log_size=10
modified=no
hsqldb.cache_version=1.7.0
hsqldb.original_version=1.8.0
hsqldb.compatible_version=1.8.0
```

The properties file contains the basic settings for the internal HSQLDB database.

database/script

```
SET DATABASE COLLATION "German"
CREATE SCHEMA PUBLIC AUTHORIZATION DBA
CREATE USER SA PASSWORD ""
GRANT DBA TO SA
SET WRITE_DELAY 60
```

The script file contains default settings for connection to the database, the language setting, etc. The user SA, to be described later, appears here.

In a database with contents, this file contains the basic table definitions.

```
SET DATABASE COLLATION "German"
CREATE SCHEMA PUBLIC AUTHORIZATION DBA
```

The tables are defined before the database user is defined. First the tables are created in the cache with their fields.

```
CREATE CACHED TABLE "Stock"
   ("ID" INTEGER GENERATED BY DEFAULT AS IDENTITY(START WITH 0) NOT NULL
   PRIMARY KEY,"Stock" VARCHAR(50),"Price" DECIMAL(8,2),"MWSt" TINYINT)
CREATE CACHED TABLE "Sales"
   ("ID" INTEGER GENERATED BY DEFAULT AS IDENTITY(START WITH 0) NOT NULL
   PRIMARY KEY,"Total" TINYINT,"Stock_ID" INTEGER,"Receipt_ID" INTEGER,
   CONSTRAINT SYS_FK_59 FOREIGN KEY("Stock_ID") REFERENCES "Stock"("ID"))
CREATE CACHED TABLE "Receipts"
   ("ID" INTEGER GENERATED BY DEFAULT AS IDENTITY(START WITH 0) NOT NULL
   PRIMARY KEY,"Date" DATE)
```

Then changes in the table are made to ensure that the relationships (REFERENCES) are consistent.

```
ALTER TABLE "Sales" ADD CONSTRAINT SYS_FK_76 FOREIGN KEY("Receipt_ID")
   REFERENCES "Receipts"("ID")
SET TABLE "Stock" INDEX'608 20'
SET TABLE "Sales" INDEX'1872 1656 1872 12'
SET TABLE "Receipts" INDEX'2232 1'
```

After setting the position of the index in the data file (it appears here only in the script file but is never actually entered directly in SQL), the automatically incrementing fields in the tables (AutoValues) are set up so that they will provide the next value on entry of a new record. Suppose the last entered value in the ID field of the Stock table is 19. Auto-incrementing then starts at 20.

```
ALTER TABLE "Stock" ALTER COLUMN "ID" RESTART WITH 20
ALTER TABLE "Sales" ALTER COLUMN "ID" RESTART WITH 12
ALTER TABLE "Receipts" ALTER COLUMN "ID" RESTART WITH 1
CREATE USER SA PASSWORD ""
GRANT DBA TO SA
SET WRITE_DELAY 60
```

Managing the internal Firebird database

The internal Firebird database is at the moment only available as an experimental function. To create such a database, or to edit one that has been created, you must select **Tools > Options > LibreOffice > Advanced > Optional (Unstable) Options > Enable experimental features**. This path illustrates well that such a database is not suitable for everyday use.

The following link allows significant bugs in the internal Firebird database to be reported and addressed jointly with the LibreOffice team: Reporting bugs for Firebird in Base.

Users will notice the following differences from HSQLDB:

1) If a field is given the type Integer and then declared as the primary key, it appears to be possible to give it an auto-incrementing value. However on saving, this setting disappears without notice.
2) When new records are entered, they are not automatically saved in the database. The Save button has to be used for each entry. In the built-in HSQLDB, the explicit saving of records is not necessary.
3) Aliases are completely ignored in queries. An alias can be created but it will not appear in the table heading of the query.
4) It is not possible to create conditions, although external Firebird databases support them.
5) The decimal and numeric data types are faulty at present. These are the only types that ensure precise values, especially when there are decimal places. They are therefore the preferred fields for currency values. At present, only values with at most one decimal place can be entered.

Making AutoValues available

The following code, entered using **Tools > SQL**, can help with the problem of auto-values not being provided.

```
CREATE TABLE "Table1" ( "ID" INTEGER NOT NULL PRIMARY KEY, "Name"
VARCHAR(20) NOT NULL );

CREATE GENERATOR GEN_T1_ID;

SET GENERATOR GEN_T1_ID TO 0;
```

After this, the SQL entry window should be closed and **View > Refresh Tables** selected. Only when the table appears, and in some cases only after an (unsuccessful) attempt to create an entry, can the following Trigger be created.

```
CREATE TRIGGER T1_BI FOR "Table1" ACTIVE BEFORE INSERT POSITION 0 AS
BEGIN
IF (NEW.ID IS NULL) THEN NEW.ID = GEN_ID(GEN_T1_ID, 1);
END;
```

Even after this, many entries in Name can be made in the table without creating an entry in ID. The ID field just shows 0. Only when **Update** is pressed are the actual assigned values displayed. The trigger provides values that begin with 1.