FREE eBook

LEARNING

ms-access

Free unaffiliated eBook created from
Stack Overflow contributors.

#ms-access

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: ms-access

It is an unofficial and free ms-access ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official ms-access.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with ms-access

Remarks

Apologies for the wall of text. Future edits will likely add screenshots and other visual elements.

Queries in Microsoft Access can be created by any one of four methods

1. Using a step-by-step Query Wizard builder through the GUI which will ask you a series of questions about what data you wish to display and how various bits of data are related to each other.
2. By using the GUI in Design View, in which you select tables and specific fields with your mouse. The visual ordering of the various fields can be specified by dragging the relevant columns in the bottom panel and additional properties for each field can be specified in the Properties panel (right).
3. By switching from Design View to SQL View and specifying the raw SQL query code. Under most circumstances you can freely toggle between Design View and SQL View - Access will incorporate changes that you made in one view to the other. The syntax of Access SQL is similar to, but not identical to, that used in MySQL, PostgreSQL, Oracle, or MS SQL Server (tSQL).
4. Using Visual Basic for Applications programming which can be accessed via the Macro group of the Database Tools ribbon (Access 2007+). Database manipulation occurs via the ADO or DAO libraries and uses the same syntax as SQL View in the main application, with the exception that various special characters have to be "escaped." Queries created via this method are not accessible directly from the Navigation Pane but must be placed in a function or subprocedure and either triggered by other elements (e.g. by a button in a Form) via Macros or directly executed in the VBA GUI interface.

Editing a record value from a query in datasheet view will result in a change in the underlying record value, assuming the query field is not an aggregation or concatenation of multiple sources of information.

Forms and Reports can be used to display information from queries in a form alternative to a simple "Datasheet" view which appears similar to an Excel-style spreadsheet. Forms are targeted to on-screen display, whereas Reports are targeted to those printed on paper.

Examples

Installation or Setup

Microsoft Access is one of the Microsoft Office suite of programs. However, it is only available in some packages of MS Office.

If you wish to obtain Access, please make sure to carefully examine the box or download specifications for each version of Microsoft Office. MS Access is **only available for Windows**

PCs, it is not available on Macintosh systems in the native environment, even through other MS Office programs may be available. Similarly, it is not available for linux operating systems.

In Office 365, Access can be found in the Home, Personal, ProPlus, Enterprise E3 or E5 versions, but **not Enterprise E1 nor Business** (or B. Essentials, B. Premium).

In Office 2016 it is **not included in the Home & Student or Home & Buisness packages**, but it is in Professional. It does not appear to be in any versions for the Macintosh.

What is MS-Access and what do we use it for ?

Microsoft Access is an **Application Generator** for developing databases and data-driven applications, primarily for local use. Microsoft Access consists of two main elements:

1. A **Relational Database Management System** (**RDBMS**) that combines the Microsoft Jet Database Engine (Access 2003 and earler) or the Access Database Engine (Access 2007 and later; see below) with graphical management tools. A unique **Linked Tables** system allows remote tables to be treated as local.
2. **Graphical User Interface** (**GUI**), and **software development** tools, supported by Visual Basic for Applications (**VBA**) that can reference a variety of objects.

It is a member of the **Microsoft Office** suite of applications, included in the Professional and higher editions or sold separately. Database applications that have been created with a full version of Microsoft Access can be compiled for distribution and run via a free Microsoft Access Runtime.

The two elements allow Microsoft Access to be used in various ways:

- **As a Database**: A Microsoft Access database does not call for a ***Data-Server***, and is often used as a database for local applications, such as a **web-site database**, located on the web-server.
- **As a Data Application Generator**: Tools for creating GUIs containing Forms and Controls bound to (*local* or *linked*) tables allow developers to create local applications for accessing and managing local or remote data. VBA modules allow developers to create capabilities not supported by GUI tools.
- **As a full Application Generator**: The above abilities allow developers to create full local data applications in one or more Access files.

Microsoft Access Database Engines

Through Access 2003 (11.0), the built-in database engine was Microsoft Jet. With Access 2007 (12.0), Microsoft introduced a new descendant of the Jet engine, the Access Database Engine (originally called the Access Connectivity Engine and still commonly known as the **ACE Engine**), and made it the default for new databases. Its feature set and behavior overlaps incompletely with the last version of Jet (4.0). Versions of Access released since have been able to create and work with databases in either Jet (`.mdb`) or ACE (`.accdb`) format, even though Jet has been officially deprecated as a technology.

Versions

Microsoft Access has existed since 1992, and older versions continue to see regular use when business-critical database applications have been built on them. A very comprehensive resource summarizing the release history (with links to release notes, where available) is:

- Microsoft Access Version Releases, Service Packs, Hotfixes, and Updates History

Read Getting started with ms-access online: https://riptutorial.com/ms-access/topic/3609/getting-started-with-ms-access

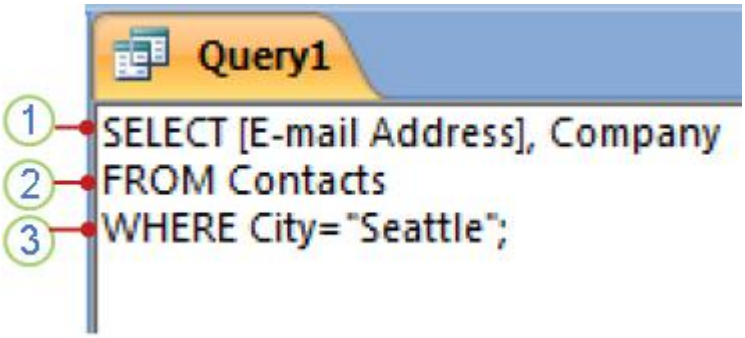
Chapter 2: Access SQL

Examples

Introduction to Access SQL

When using Access you can retrieve data using queries. These queries are built using Structured Query Language (SQL). Understanding SQL is important because it can help build better, more useful queries.

When creating queries in Access, you can switch to "SQL View". An example of a "select" query is shown here:



Union (Merge) Queries

When you wish to combine the results of multiple tables or queries with similar fields together into a single resulting data set without performing any relational joins (i.e. you want to list one dataset immediately after the other), you will use a `UNION` query. However, it is notable that **these queries must be manually created in SQL View.**

Syntax of a `UNION` query is

```
SELECT
  floatingpoint_field AS floatptfld,
  text_field
FROM first_table
  UNION
SELECT
  integer_field,
  decimal_field
FROM a_saved_query
  UNION
SELECT
  1.0,
  "hi there Jack"
```

and will return a two-field dataset with field (column) names: `floatptfld` and `text_field`

It is critical that the data types (and data styles) for subsequently merged tables fields are compatible with the first query in the series. In other words, if the first `SELECT` query generates a

number for the first column, the second query must also return a number in the first column. In addition to matching types of fields in order, the `SELECT` statements must return the same number of fields. Names for the fields of the resulting datasheet are inherited from the first table definition.

The following query would NOT be legal, as text cannot be turned into decimal data nor can floating point numbers be converted to integers (without explicit truncation or rounding and type-casting).

```
SELECT
  integer_field AS this_really_wont_turn_out_well,
  decimal_field
FROM a_saved_query
  UNION
SELECT
  floatingpoint_field,
  text_field
FROM first_table
```

The COUNT() Function

You can use the COUNT() function to return the number of records that match a query. The following 'Employee' table contains employee ID numbers and their associated manager's ID number.

Employee_ID	Manager_ID
12	37
22	37
37	63
42	45
45	63
57	45
59	45
63	

A COUNT() statement can be used to find out how many employees have a specific manager:

```
SELECT COUNT(*) AS CNT FROM Employees WHERE Employee.Manager_ID = 37;
```

returns

CNT

2

The function can also be combined in more complicated queries. To find out how many employees are directly supervised by a specified person, the following can be applied:

```
SELECT T1.Employee_ID,
    (SELECT COUNT(*) AS CNT FROM Employees AS T2 WHERE T2.Manager_ID =
       T1.Employee_ID) AS Supervised_Count
FROM Employees AS T1;
```

returns:

Employee_ID	Supervised_Count
12	0
22	0
37	2
42	0
45	3
57	0
59	0
63	2

MSDN documentation may be found here.

Read Access SQL online: https://riptutorial.com/ms-access/topic/3985/access-sql

Chapter 3: How to troubleshoot Access crashes

Introduction

When you receive an error: "Microsoft Access has encountered a problem and needs to close", there is often not a lot of information to help you identify the cause of the error. Below are a series of steps you can take to troubleshoot the cause of the errors.

Remarks

Be sure to remove other variables from the equation while testing

Network Corruption

Do not load the client off of a network. Put it on the local drive and run it from there.

Corporate Builds

If you are in a corporate environment that is using "computer builds" and have had no success with Decompiling, Testing Memory, nor stripping Binary data - then refuse to do further testing until the IT team can provide the user with a test machine that has only Windows, Office, and Service Packs installed.

All software and updates should be installed by hand without using unattended installs. Do not install antivirus on this machine for testing.

Understand that many IT departments simply try to do a One-Size-Fits-All approach with the builds, and their builds are all based on one another. Over time, software conflicts can directly cause Access to crash or act strange.

Bad Power

As mentioned in the memory example - power fluctuations can cause computer errors. If the database is in an industrial building, then try to get your hands on a power conditioner or a UPS that provides clean power (off the battery, not off the main passing through a Metal-oxide Varistor)

Also, check the power supply cable that is plugging into the power bar or outlet. Make sure that the gauge and voltage specs are sufficient. IT departments often leave power cables plugged in at the station and just remove the machine. After many years, they are using beefier power supplies, but haven't switched out the cable. It makes a difference. When in doubt, bring a new, thicker cable.

Examples

Decompile Database

This should always be your initial fix. A good policy is to decompile the database before each release.

1. **Create a decompile shortcut**. This loads the database with a "/decompile" switch.

    1. Right Click your Database File. Select Copy
    2. Right Click in the explorer window and select "Paste Shortcut"
    3. Right Click the shortcut and select "Properties"
    4. In the Target box, go to the end of the line and add `/decompile`
    5. Click Ok to close the shortcut

2. **Open Database with Shift.**
   Hold down the shift key while double clicking on this shortcut. This prevents any auto runs from executing within the database. You should go straight to the navigation window.

3. **Compact and Repair the database.** Once the database is loaded, you will need to click the Compact and Repair button.

    1. Locate the *Compact and Repair Database* button on the **Tools** Ribbon.
    2. Hold down the Shift Key. Keep holding it down while you click the *Compact and Repair* button.

4. **Recompile the Database**

    1. Go into the VBA window (Control + G)
    2. Select Debug -> Compile from the menu

This is the complete decompile process. Generally it should fix 99% of all Access crashes or weird form behaviour.

Test Computer Memory

If your crashes are random or sporadic, then do this step. If your crashes occur every single time you run the database, then this step won't fix the issue (although bad memory may be the reason why the corruption occurred in the first place).

Use a memory tester that boots outside the operating system and runs multiple passes. Two popular choices are MemTest86 (Commercial) and MemTest86+ (Open Source)

Start the test and let it run during working hours. The reason for this is because other factors in the building such as noise on the power circuits can cause memory errors, so you want to try to keep the variables the same.

If you have memory errors, then you will need to identify whether it is due to bad memory in the computer, or some other factor. This is beyond the scope of this document however.

Remove Binary Data from Form

Sometimes the crashes occur constantly in a single form or report, or occur only when printing. It is possible that the binary data within the form / report has become corrupt.

Save the Form / Report object as text There are two undocumented functions. Application.SaveAsText and Application.LoadFromText. You can use these functions to export the form/report definitions, clean up the definition, and then import it again.

1. Make a backup of your database before proceeding
2. Go to the VBA Immediate Window (Control + G)
3. Type `Application.SaveAsText acForm, "MyForm", CurrentProject.Path & "\MyForm.txt"` (Replace MyForm with the name of the Form / Report. Use acReport if it is a corrupt report you are fixing)
4. Rename the original form item (e.g. rename to MyForm.Bak) within the Database Window

Clean up the Form / Report Definitions file

1. Open the exported file (e.g. MyForm.txt) in notepad

2. Delete the "Checksum=" line (should be on line 3)

3. Clear out binary data

    1. Identify the binary data blocks. Look through the file and you will see lines that start with "Parameter = Begin". Following those lines you will have lines of encoded binary data. Finally, the binary block will end with a line consisting only of "End". The Binary Data block includes the first line (with the Begin statement) and all lines up to and including the last line (with the End Statement).

        Note: All of these blocks should appear BEFORE your form control definitions

    2. Delete the binary data blocks for the following parameters:

        - NameMap
        - PrtMip
        - PrtDevMode
        - PrtDevNames
        - PrtDevModeW
        - PrtDevNamesW

4. Look for other issues. While you have the file open, scroll through the rest of the file and look for anything that catches your eye, especially in the VBA module code at the bottom. You will be looking for anything that sticks out from the rest, and may be corruption.

5. Save the file.

Load the form / report back into Access and Test

1. Load the form back into Access.
    - In Access, go to the immediate window (Control + G)
    - Type `Application.LoadFromText acForm, "MyForm", CurrentProject.Path & "\MyForm.txt"`

- Decompile / Compact Repair / Recompile (See the other example within the documentation)
- Open the form / report to test. Hopefully everything is working now.
- Delete the old corrupt form (e.g. MyForm.bak)

Prevent this corruption in the future

The most common cause of corrupt binary data within a report / form is when multiple computers / users use the same database client file instead of having their own separate copy. This is why each user should have their own client file on their desktop that they run.

Remove "OLE Object" fields

If you have images or other data stored in Access itself as OLE Objects, then you should find a better approach. When the OLE data is stored, it is stored according to the software (and version of software) on the computer storing it. When another computer goes to display that OLE Object data on the form, but doesn't have the exact software / version installed - quite often this results in an application crash.

If you are storing image data, then a better approach is to store the filename, and instead save the images to a standard location. Newer versions of access have the native controls to make this the way to go.

Rebuild the entire database

This is a lot of work, so do this as a last resort after exhausting all other options. You only need to do this if the problem is occurring for different users, on different machines. If it isn't occurring for all users, then most likely it is not a corrupt database container.

Similar to the steps in removing binary data, you are going to rebuild your database from scratch. This process is a little ritualistic, but if done meticulously with care not to "preserve" any possible corruption, then the process is highly effective.

Create a new access database container.

- In Access, on the File Tab, you can select "New". Create a new, empty database in ACCDB format.

Move all objects to the new container

Do **not** use the Import / Export functions within Access to move the objects, and do not simply click and drag. Doing this can copy the corrupt items over to the new container.

Tables:

- For each table in the old access container, create a new table in the new container.
- From design view, copy/paste the field definitions.
- Check the table properties to ensure they match in both databases
- Move any Data Macros over as well (see the Macros section for how to do this)

- To move the data, export the old data to XML or CSV, and then import from that format.

Queries:

- Load each query into SQL view.
- Copy / Paste the SQL text.
- Paste into the new database.
- Compare Query properties to ensure they match.

Forms / Reports:

- For each Form / Report, use the Application.SaveAsText function to export the forms/reports to a text file.
- Remove the Binary Data (see *Remove Binary Data from Form* documentation to acquaint yourself with this process)
- Use the Application.LoadFromText function to reimport the objects into the new database

Macros

You have three methods of moving the Macros.

1. Recreate each macro by hand in the new database container.
2. Use the `Application.SaveAsText` / `Application.LoadFromText` method with the `acMacro` parameter.
3. Copy/Paste Macro definitions for each macro
    - Select All (Control + A) to select all macro elements. Then Copy (Control + C).
    - Open a blank Notepad document and Paste (Control + V) the Macro XML.
    - Create a new blank macro in the new database container.
    - In Notepad, Select All text (Control + A). Then Copy (Control + C)
    - In the blank macro, Paste (Control + V). The Macro should appear. Save it.

Modules

- For each module, select all code (Control + A) and paste (Control + V) into the new database container.
- Be sure to check the Database Properties (In VBA Window, go Tools -> Client Properties)

Data Macros

For each Data Macro, use the SaveAsText / LoadFromText methods.

1. Go into the VBA Immediate Window (Control + G)
2. Type `Application.SaveAsText acTableDataMacro, "MyTableName", CurrentProject.Path & "\MyTableName.txt"` (Replace MyTableName with the name of the table containing the data macros)
3. Review the file for any signs of corruption
4. In the new database container, load the definition using `Application.LoadFromText acTableDataMacro, "MyTableName", CurrentProject.Path & "\MyTableName.txt"`

As previously mentioned, this is a **LOT** of work, but it has results. This method should also be

used when migrating an Access 97 database to 2000, or an Access 2000 database to 2003.

Read How to troubleshoot Access crashes online: https://riptutorial.com/ms-access/topic/8207/how-to-troubleshoot-access-crashes

Chapter 4: Parameterized Queries

Introduction

Parameterized Queries can be used to defend against SQL Injection attacks.

Examples

Vulnerable Approach: Concatenated SQL string with form references

This is the typical approach for novice developers building SQL action queries. They are vulnerable to the Bobby Tables type SQL Injection attacks.

```
Dim strSQL As String

strSQL = "INSERT INTO Employees chrFirstName, chrLastName, chrPhone " _
         & "VALUES ('" & Me!txtFirstName & "','" & Me!txtLastName & "','" & Me!txtPhone &
"');"

CurrentDb.Execute strSQL
```

QueryDef Parameterized Query Approach

This approach will prevent a user from embedding a second SQL statement in their input for execution.

```
Dim strSQL As String
Dim db As DAO.Database
Dim qdf As DAO.QueryDef

strSQL = "PARAMETERS [FirstName] Text(255), [LastName] Text(255), [Phone] Text(255); " _
         & "INSERT INTO Employees (chrFirstName, chrLastName, chrPhone) " _
         & "VALUES ([FirstName], [LastName], [Phone]);"

Set db = CurrentDb

Set qdf = db.CreateQueryDef("", strSQL)
qdf.Parameters("FirstName") = Me!txtFirstName
qdf.Parameters("LastName") = Me!txtLastName
qdf.Parameters("Phone") = Me!txtPhone
qdf.Execute

Me!txtFirstName = vbNullString
Me!txtLastName = vbNullString
Me!txtPhone = vbNullString

qdf.Close
db.Close
Set qdf = Nothing
Set db = Nothing
```

Valid Parameter Types:

- `DATETIME`: for dates (parameter expects VBA `Date`)
- `SHORT`,`LONG`: For integers (`SHORT` expects Integer, `LONG` expects Long)
- `SINGLE`,`DOUBLE` : For floating point (expect Single and Double respectively)
- `VARCHAR` or `TEXT`: For strings
- `MEMO` or `LONGTEXT`: For strings longer than 255 characters

Read Parameterized Queries online: https://riptutorial.com/ms-access/topic/9121/parameterized-queries

Chapter 5: Self-Referencing tables

Remarks

In the example above, a reference field (SupID) can be used to indicate the ID of that employee's supervisor.

Using something as simple as a DLOOKUP can return the name of that supervisor.
eg. `DLOOKUP("Name","EmployeeTable", "ID = " & SupID)`

Another good example of this is to look at how automated Access switchboards are created, and more specifically the construct of the Switchboard table. Each switchboard option refers to another option within the same table - similar to how this example self references.

Examples

Self Referencing Employee Table

ID	Name	SupID
1	Jim Browski	3
2	Sally Valley	3
3	Boss Hawg	
4	Yvette Deer	6
5	Mary C Lyte	6
6	Chuck Dee	
7	Rick Slick	6
8	Doug Fresh	3

Read Self-Referencing tables online: https://riptutorial.com/ms-access/topic/7992/self-referencing-tables

Credits

S. No	Chapters	Contributors
1	Getting started with ms-access	Brad, Community, Hynek Bernard, jcb, marlan, mpag
2	Access SQL	LiamH, mpag, SandPiper
3	How to troubleshoot Access crashes	DHW, user3728595
4	Parameterized Queries	MoondogsMaDawg, serakfalcon
5	Self-Referencing tables	geeFlo