

# EducAide Software

---

## Instructions for producing an Acces-compatible database module (program version 3.41 or higher)

### Requirements

In order to produce your own database module, you need:

- a license for *Acces*, which means you must be an employee at an elementary or secondary school that is site-licensed;
- *Acces* for Windows version 3.41 or higher (you will meet this requirement if you installed from a CD labeled 1.2x);
- the *Acces* program installed on your computer and running properly;
- experience writing problems in *Acces* and working with the T<sub>E</sub>X language;
- a basic understanding of DOS (how to change directories, do a file listing, and run command-line utilities); and
- a good text editor, such as one for programming or Web authoring, and practice using it.

Also note:

- If you do not have a text editor, then you can use NotePad or WordPad, which come with Microsoft Windows<sup>tm</sup>. But be forewarned: those program are cumbersome for editing files that do not have the extension TXT, and they lack good automation, multi-file editing, and search and replace capabilities.
- If your version of *Acces* is older than 3.41, then you will need to obtain an updated CD. There may a small charge depending on when you purchased the software. Please contact EducAide for details.
- If your version of *Acces* is 3.41–3.46, then you need to update a few files. If you have not done so already, save the file DBC001.EXE in your main *Acces* directory (folder), then go to a DOS prompt, change to that directory, and give the command: `DBC001 /D /N`

Important! Do not try to update a version of *Acces* older than 3.41 with the file DBC001.EXE. Doing so will make *Acces* unusable, and you will have to re-install it.

## Overview

When producing your own database module for *Acces*, you go through the following six steps (not always in the same order):

- Create “raw” or uncompiled data files. These are ASCII text files that are named and structured a certain way, contain your original problems, and correspond to the categories in your database module.
- Describe your categories and assign default directions and workspace in a “database description file”. This is an ASCII text file that is named and structured a certain way and is unique for each module.
- Verify your data files. This means checking the file structure and the coding of your problems (TeX syntax).
- Compile your data files into an *Acces*-compatible module.
- Produce a catalog that shows all of the problems in your module.
- Install your module into *Acces*. This step may be as simple as copying two files and adding the database code to *Acces*’ registry.

If you previously worked with *Extend* (a program that used to ship with *Acces* but is now obsolete), then you will be quite familiar with the above steps. You should read the section called “Raw data files” to learn about a few minor changes. Otherwise, any files you created in the past should compile just fine. The main differences that you should be aware of are:

- Syntax checking, catalog production, and database compilation are now done on the command line.
- Categories are described in a text file, rather than on a grid in the *Extend* program. The text file is named xxx.DB, where xxx is the database code.
- All problems in a data file should be delimited by curly braces, not with `\plines` and `\alines` commands. While those commands are still supported, *EducAide* intends to phase them out (along with `\pmode` and `\amode`).
- It is no longer necessary to make a database code with DBUTIL. Now all development is separated from *Acces*, and *Acces* needs to “know about” a module only when it is time to select problems from it.

If you never worked with *Extend*, then you need to be aware of the following:

- When you produce a database module for *Acces*, you assign it a database code. The code must be similar to those used for *EducAide*’s modules—that is, it must be made up of three alpha-numeric characters, such as TX3. The main restriction is that your code must start with a numeral, such as 1AB. This prevents conflicts with *EducAide*, whose codes always start with a letter.
- In our examples, we will be referring to the EXT or “Extra” database. This is a small database module that ships with *Acces* and is distributed with its “raw” or uncompiled data files. You are welcome to modify the files and use the code for your own purposes. (This is the only database you can compile whose code starts with a letter.)

## Sample session

Here we guide you through the steps of producing a small database module. This will help you understand the process, then you can read the sections that follow for more details. Before continuing, please be sure that you have met the requirements listed above and you have updated your version of *Acces*, if necessary.

**Step 1.** Begin by looking at the problem catalog for the EXT (“Extra”) database. This is a small pink booklet that is included with *Acces* and referred to in the Reference Manual (30-minute tutorial).

If you do not have a printed copy of the EXT catalog, then you can look at the electronic version. In Windows, open your *Acces* program folder (normally C:\ACC), then open the folder called “Catalogs”. Double-click the file EXT.PDF. You can view the EXT catalog with Adobe Acrobat™ Reader and re-print some or all of the pages.

Notice the table of contents at the front of the EXT catalog. There are 13 categories, AA–AM. These cover various typesetting techniques, mostly for mathematics.

**Step 2.** Now go to a DOS prompt and change to your *Acces* directory, normally with the command CD \ACC. Then give these two commands, which will move you to a subdirectory of *Acces* called “Samples” and list the files:

```
CD SAMPLES
DIR /ON
```

You should see these files, plus any that you may have created in the past:

```
Directory of C:\Acc\Samples

AA      EXT      4,525  01-14-02  3:40p
AB      EXT      3,939  01-14-02  3:40p
AC      EXT      2,826  01-14-02  3:40p
AD      EXT      2,596  01-14-02  3:40p
AE      EXT      5,676  01-14-02  3:40p
AF      EXT      4,031  01-14-02  3:40p
AG      EXT     10,004  01-14-02  3:40p
AH      EXT      4,459  01-14-02  3:40p
AI      EXT      2,343  01-14-02  3:40p
AJ      EXT      2,787  01-14-02  3:40p
AK      EXT      5,990  01-14-02  3:40p
AL      EXT      5,970  01-14-02  3:40p
AM      EXT      5,108  01-14-02  3:40p
EXT     DB         615  01-14-02  3:40p
EXT     TOC         594  01-14-02  3:40p
```

The above files are used to build the EXT database. You can clearly see the naming conventions. For data files, the name is the category code (in this case, AA through AM) and the extension is the database code (EXT). For the “database description file”, the name is the database code and the extension is DB. (The file with the extension TOC is for creating a table of contents. For more information, see the section called “Catalog production”.)

**Step 3.** Open the data file AA.EXT in your text editor. If you don't have an editor, give the command: `NOTEPAD AA.EXT`

Notice the lines that begin with a percent (%) symbol. In the  $\text{\TeX}$  language, the percent is a comment character; it is used to hide notes or embed comments in ordinary text. We follow that convention with data files: any line that begins with a percent will not be printed, and, if the line is outside a problem or answer block, it will not even be compiled into the database.

As you scroll through the file, you will see the problems in category AA delimited by curly braces. You will also notice the answers follow immediately, always on a new line. By default, a problem or answer is expected to take up one line. If either takes up more (a problem usually requires several lines), then it must be "delimited" with curly braces.

Probably you know that curly braces are an integral part of the  $\text{\TeX}$  language. They are used as grouping symbols, to delimit the arguments of a command, etc. For that reason, we have adopted them as delimiters within a raw data file. So that there is never any confusion about whether a brace delimits a problem or is part of it, there is a simple rule to follow: if a curly brace delimits a problem (or answer), then it should be on its own line, with no other characters and no indentation.

**Step 4.** Now close the file AA.EXT and open the database description file, EXT.DB. (If you are running Windows Notepad, the easiest method is actually to quit the program and re-start it with the new file name.)

Notice the first four lines contain general information about the database. This is a requirement. In the DB file:

- Line 1 contains the database code
- Line 2 contains a brief description
- Line 3 contains a copyright or other notice
- Line 4 contains the direction file name, normally ACCES.DIR

Subsequent lines describe the categories. Each line contains, in this order:

- a two-letter category code
- default direction number
- default workspace value (tenths of an inch)
- brief description of the category (optional)

Obviously, the codes tell the database compiler which categories (files) to include. The codes should be in alphabetical order mostly for the purpose of producing a problem catalog, but the categories need not be complete or consecutive (i.e., category codes can be skipped).

The direction numbers and workspace values show up in *Acces'* grid, as defaults, when a problem is selected from the database. Of course, they can be overridden at that time.

The descriptions are optional; they show up only in the printed problem catalog. If you look at the EXT catalog again, you'll see the descriptions at the beginning of the categories, each labeled as a "memo".

**Step 5.** Now close the file EXT.DB and quit or minimize your editor. You are ready to compile the database.

Note: In the steps that follow, we assume that the *Acces* program directory is C:\ACC. If not, please substitute the name of your directory when giving a command.

While you are still in the Samples directory, give the command:

```
C:\ACC\DBCMP /X EXT
```

You just compiled the database! Since it has less than 200 problems, the compilation was probably over before you knew what happened. In fact, even with thousands of problems, compilation does not take more than a few seconds.

The main reason for doing the compilation is turn text files (“raw data”) into binary files that can be indexed and read more quickly by *Acces*. There is also a hidden operation: a check on the structure of the files and the validity of the code—that is, T<sub>E</sub>X syntax. In the case of EXT, the files were sent to you error-free. (Well, they are error-free in terms of file structure and code; there may be typos or math mistakes that we never caught).

The database compiler is concerned with two things mainly: curly braces and dollar signs must be matched within each problem (and answer), and items must be delimited properly in the data file. If there are any errors, you will be notified. In that case, the DBCOMP program will end normally, but it will NOT produce a compiled database.

**Step 6.** Now that you have run the DBCOMP program, do a directory listing with the command: DIR EXT.\*

In addition to the files that were listed before, you should see:

EXT	CFG	527	01-30-02	1:53p
EXT	DAT	12,793	01-30-02	1:53p
EXT	REP	972	01-30-02	1:53p
EXT	TEX	2,757	01-30-02	1:53p

All four of those are “output files”. They were created by DBCOMP in the previous step. The first is an index or configuration file for *Acces*, and the second is compressed, binary data. Neither EXT.CFG or EXT.DAT is “human readable”, so you can ignore those files for now.

Very important: DBCOMP will never overwrite or erase a DB file or any raw data files that you create; it reads them only. Also, you can re-run the program as often as you like; it simply regenerates the output files.

**Step 7.** Open the file EXT.REP in your editor. This is a “report file”. It summarizes the contents of the database and lists any errors that may have occurred during compilation. (If there were errors, they would be listed by file name and line number, making it easy for you to correct them.)

**Step 8.** Now close the file EXT.REP and open the file EXT.TEX. This is a “master T<sub>E</sub>X file” for the problem catalog. It was created with an optional switch when you ran the DBCOMP program (/X). If you scroll through the file, you will see some general information about the catalog at the beginning, then a `\begindocument... \enddocument` group for each category. Of course, all of the coding in this file is for the T<sub>E</sub>X typesetting system.

One thing you will not see in EXT.TEX are problems and answers. How can that be? The reason is that raw data files (categories) are read “on the fly”, when a catalog is being typeset. Thus, you do not need to think about re-creating the master T<sub>E</sub>X file unless category information changes. In fact, re-creating the file may never be desirable, because you may want to edit it and customize the look of your problem catalog. (You should edit the file only if you are familiar with T<sub>E</sub>X and the ACC format.)

**Step 9.** Now close the file EXT.TEX and quit or minimize your editor. It is time to make the problem catalog.

While you are still in the Samples directory, give the command:

```
C:\ACC\VTTEX3S &ACC EXT
```

The previous command will launch the T<sub>E</sub>X typesetting system, in much the same way as it is launched in *Acces*. The main difference is that messages are not hidden from you; they appear in the DOS window. If you encounter an error, it will probably be due to you’ve coded a problem or answer. Although T<sub>E</sub>X’s error messages can be difficult to interpret, you should be able to discern the file name and the “offending” command. If an error message scrolls off the screen too fast, you can see it again by opening the file xxx.LOG, where xxx is the database code.

**Step 10.** When T<sub>E</sub>X is finished, you can preview (and optionally print) the catalog. Give the command:

```
C:\ACC\VTTEXWD EXT
```

You should recognize the previewer from *Acces*. You should also recognize the EXT problem catalog inside the previewer. Except for the cover and table of contents, which we will discuss later, the catalog should appear the same as the PDF file or “hard copy” that you looked at earlier.

When you are finished previewing the catalog, press ESC or choose Exit from the File menu.

**Step 11.** The last step is to install the EXT database into *Acces*. (In this case, the database is already installed, so it is more a matter of updating the files.)

For *Acces* to recognize a database module,

- there must a subdirectory (below the main *Acces* directory) whose name is the database code,
- the CFG and DAT files (which are output by DBCOMP) must be copied to that subdirectory, and
- the database code must be added to *Acces*’ “registry”.

To make the directory for EXT, you would give the command:

```
MD C:\ACC\EXT
```

You can try that now; you'll get a message saying that the directory already exists.

To copy the necessary files into the directory, give the commands:

```
COPY EXT.CFG C:\ACC\EXT
COPY EXT.DAT C:\ACC\EXT
```

When you are asked if you want to overwrite the existing files, answer yes.

Note: In the directory C:\ACC\EXT, you may notice files called EXTU.CFG and EXTU.DAT. These were for the old Extend program. They are no longer needed, and you can delete them after copying the new CFG and DAT files.

Finally, to add the code to *Acces*' registry, you would give the command:

```
C:\ACC\DBUTIL /A EXT
```

You can try that now; you'll get a message saying that the database is already installed.

```
* * * * *
```

The sample session has now come to an end. You may work some more with the EXT database or start creating you own data files. In the sections that follow, you will find many hints and shortcuts for compiling a database. In particular, if you have already written problems in *Acces* that you want to use, please see the section called "Exchanging data with *Acces*".