Current News

22 November 2016 10 reasons to turn your Access applications into Web-based applications

When an Access database outgrows its original purpose, you face applying band-aid technology or upgrading to a more powerful database system. But before you toss Access out the window and start signing purchase orders for consultants, developers, licensing, and new hardware, consider one more option -- turning your Access application into a Web-based application. Here are a few reasons why this might make sense.

An Access database often outgrows its original purpose. When that happens, you face applying band-aid technology or upgrading to a more powerful database system, such as SQL Server Express or even SQL Server. But before you toss Access out the window and start signing purchase orders for consultants, developers, licensing, and new hardware, consider one more option — turning your Access application into a Web-based application. Let's look at some reasons why this might make sense.

#1: Client versus server

A server-side database, such as MySQL, SQL Server, and Oracle, evaluates requests on the server side (sent in the form of a SQL statement) and then returns data to the client. Jet, on the other hand, lets the client do all the work. Jet is the database engine behind Access. Even if the database (.mdb) is on a network server, the client still does all the work. The server simply responds to client file requests.

This arrangement retrieves more then just the data across the network. As a result, indexes and unused data clog the network and slow things down. An alternative is to place the Access database on your Web server's local drive and then build the interface on the Web server. Doing so creates an ad hoc server-side database that handles transactions on the server (using your code). Requests from the client are in Hyper Text Transfer Protocol (HTTP) format instead of SQL.

#2: No client installation

A Web-based front end minimizes installation issues. Users need only a browser. The database doesn't care whether the user is sending requests via a Windows PC, a Mac, or a machine running Linux.

#3: Easy cross-platform usage

You're free to use your language of choice to create the Web interface and the code that the server users to interact with the database. Users get clean and standard HTML that almost all browsers can use.

#4: Simplified security

Antrow software development
Am Galgenberg 15
DE-89407 Dillingen
Germany
0176 30398469
support@antrow.com
https://www.antrow.com

Storing the database in a non-shared folder (see #1) restricts access. Only the Web server's administrator has access to the database file. That leaves security to the Web server. Now, you might argue either way as to whether this method is more or less secure than a server-side database. However, someone with direct access to a machine with a server-side database could probably also gain direct access to that database.

In addition, a server-side database requires a network connection. An Access database on a Web server isn't directly available. You can access the Web server, but not the database. Only the Web server can access the database on the server's local drive. On the other hand, Access has a security system known as Access User Level security (this isn't available with Access 2007). Most server-side database security systems are more secure than Access User Level security.

#5: Easy use of NT authentication

Using Visual Basic for Applications (VBA), you can determine the NT name of users logged into an Access database and thereby restrict which users can do what. However, this method isn't foolproof, and it doesn't truly authenticate users. Your Web interface (on an IIS Web server) can use Integrated Windows Security to authenticate user credentials to individual web pages.

#6: Goodbye to corruption!

Most developers complain that Access is susceptible to corruption. Used incorrectly, it certainly is. With an Uninterruptible Power Source (UPS) and redundant drives, your Web-based database (.mdb file) won't suffer from corruption.

#7: No version problems

With the quick pace of upgrades, many of us have users spread across two and three versions of Access. Unfortunately, not all versions play well together. A Web interface eliminates version incompatibility issues because the Web server uses Jet. That means the Web server doesn't even need Access — it doesn't load Access. Your Web server doesn't care what version of Access the client uses.

#8: Live, behind-the-scenes interface updates

To update an Access front end, you must copy or modify an .mdb file. Access won't let you make changes while people are using it. (Beginning with Access 2000, you can make some changes, but a few still require exclusive access to the database.) In contrast, you can change the Web interface files (.asp, .aspx, and so on) whenever you like. The changes are almost immediate.

#9: Portability

Every Windows OS since Windows XP has had personal Web server capabilities. That means you can develop and test a Web site using a laptop running Windows 98 (or later). Using an Access database as the data source has a few benefits:

There's no need to install and run a heavy-duty server-side database on your laptop.

There's no need to maintain a network connection to a live server.

You can copy the live system and its database as just a bunch of files. You don't have to import, export, or attach database files. For example, you can build a Web site on your laptop or desktop and then move it to a Web server. To work on an update, simply copy the Access database file (.mdb) from the Web server to your laptop. Recommendation: Jet allows many transaction type SQL statements. You can build and modify tables and views using SQL, along with the typical data reading and altering capabilities. Sometimes, if you put a system on a remote server where you no longer have the ability to get to the actual .mdb, it's pretty simple to whip up an .asp page that lets you run SQL on the fly against the database.

#10: More users

By their very nature, Web interfaces are unbound. In other words, once a page is loaded, the interface is no longer connected to the database. But a bound Access front end maintains a connection to the source, and Jet limits you to 255 concurrent connections. Your Web application, unless you have 255 users hitting the database at the exact same moment (which would require approximately 30,000 users a minute at a transactions speed of .5 seconds) can have more concurrent users.

By Susan Harkins and Drew Wu Staff