What are the Microsoft Data Access Components?

The Microsoft Data Access Components (MDAC) are the key technologies that enable Universal Data Access. Data-driven client/server applications deployed over the Web or a LAN can use these components to easily integrate information from a variety of sources, both relational (SQL) and non relational. These components include Microsoft ActiveX Data Objects (ADO), OLE DB, and Open Database Connectivity (ODBC).

ActiveX Data Objects (ADO)

ADO is the strategic application programming interface (API) to data and information. ADO provides consistent, high-performance access to data and supports a variety of development needs, including the creation of front-end database clients and middle-tier business objects that use applications, tools, languages, or Internet browsers. ADO is designed to be the one data interface needed for single and multi-tier client/server and Web-based data-driven solution development. The primary benefits of ADO are ease of use, high speed, low memory overhead, and a small disk footprint.

ADO provides an easy-to-use interface to OLE DB, which provides the underlying access to data. ADO is implemented minimal network traffic in key scenarios, and a minimal number of layers between the front end and data source-all to provide a lightweight, high-performance interface. ADO is easy to use because it uses a familiar metaphor-the COM automation interface, available from all leading Rapid Application Development (RAD) tools, database tools, and languages on the market today. ADO is a nice wrapper for OLD-DB.

OLE DB

OLE-DB is the Microsoft strategic system-level programming interface to data across the organization. OLE DB is an open specification designed to build on the success of ODBC by providing an open standard for accessing all kinds of data. Whereas ODBC was created to access relational databases, OLE DB is designed for relational and non relational information sources, including mainframe ISAM/VSAM and hierarchical databases; e-mail and file system stores; text, graphical, and geographical data; custom business objects; and more.

OLE DB defines a collection of COM interfaces that encapsulate various database management system services. These interfaces enable the creation of software components that implement such services. OLE DB components consist of data providers, which contain and expose data; data consumers, which use data; and service components, which process and transport data (such as query processors and cursor engines). OLE DB interfaces are designed to help components integrate smoothly so that OLE DB component vendors can bring high-quality OLE DB components to market quickly. In addition, OLE DB includes a bridge to ODBC to enable continued support for the broad range of ODBC relational database drivers available today.

Open Database Connectivity (ODBC)

The ODBC interface is an industry standard and a component of Microsoft Windows Open Services Architecture (WOSA). The ODBC interface makes it possible for applications to access data from a variety of database management systems (DBMSs). ODBC permits maximum interoperability-an application can access data in diverse DBMSs through a single interface. Furthermore, that application will be independent of any DBMS from which it accesses data. Users of the application can add software components called drivers, which create an interface between an application and a specific DBMS.

When should you use OLE-DB, ADO, DAO, or ODBC ?

ADO is a wrapper around OLE-DB so you can use ADO or OLE-DB.

Non-OLE environment : If a database supports ODBC and and that database is on a server that don't support OLE then ODBC is your best choice.

Non-SQL environment : ODBC is designed to work with SQL. If you have non-SQL environment then OLE-DB is better choice.

OLE environment : If you already have ODBC drives then you can use ODBC, otherwise use OLE-DB.

Interoperability required : If you need interoperable database components, then OLE-DB is your best choice.

16-Bit data access support : ADO don't support 16 bit so ODBC is the only choice.

Using multiple databases - If you are using databases that support Microsoft's jet engine then definite choice is ADO or DAO. By using ADO you get workspace level support for transaction. That means you can connect more than one database at a time in an application, which is impossible by using ODBC. You can only connect one database at a time by using ODBC.

ADO vs. DAO

ADO is a superset of DAO in functionality point of view. In fact ADO is a combination of DAO + RDO. I would prefer ADO in these cases. 1. If you are proficient in COM programming. 2. If your server supports OLE environment. 3. If you want workspace-level and multi database type (Relational, indexed, ISAM type, text files supports). DAO is best choice when you have DAO components installed on your machine and you are using Microsoft's jet database engine based databases such as MS-Access, SQL

Server, MS-Excel or Paradox. This is fastest and easiest method to access databases.

DAO vs. ODBC

DAO is best choice when you have DAO components installed on your machine and you are using Microsoft's jet database engine based databases such as MS-Access, SQL Server, MS-Excel or Paradox. This is fastest and easiest method to access databases. ODBC is for various type of database which provides ODBC drives such as SQL Server, Oracle, MS-Access. Additional advantage is DAO is workspace-level support.

Advantages of DAO: Easy to use. Workspace level support. Both MFC and API provides DAO support. Speed is relatively faster than ODBC for jet database engine databases.

Advantages of ODBC : Easy to use specially when developer are not familiar with COM environment. SQL support. Both API and MFC support. Good for relational databases only.

Advantages of ADO or OLE-DB : Workspace-level support. Fastest method to access various kind of databases. Easy to use fi you are familiar with COM environment. Provides access to relational, non relational and other types of data.