**Chapter 14 - Review Questions**

**1. Give some examples of database connectivity options and what they are used for.**

* Database connectivity refers to the mechanisms through which application programs connect and communicate with data repositories. The database connectivity software is also known as database middleware, because it represents a piece of software that interfaces between the application program and the database. The data repository is also known as the data source, because it represents the data management application (i.e. an Oracle RDBMS, SQL Server DBMS, or IBM DBMS) that will be used to store the data generated by the application program. Ideally, a data source or data repository could be located anywhere and hold any type of data. For example, the data source could be a relational database, a hierarchical database, a spreadsheet, a text data file, and so on.

**2. What are ODBC, DAO, and RDO? How are they related?**

* Open Database Connectivity (ODBC) is Microsoft’s implementation of a superset of the SQL Access Group Call Level Interface (CLI) standard for database access. ODBC allows any Windows application to access relational data sources using SQL via a standard application programming interface (API). ODBC was the first widely adopted database middleware standard and enjoyed rapid adoption in Windows applications. As programming languages evolved, ODBC did not provide significant functionality beyond the ability to execute SQL to manipulate relational style data. Therefore, programmers needed a better way to access data. To answer this need, Microsoft developed two other data access interfaces:

1. **Data Access Objects (DAO)** is an object oriented API used to access MS Access, MS FoxPro and dBase databases (using the Jet data engine) from Visual Basic programs. DAO provided an optimized interface that exposed the functionality of the Jet data engine (on which MS Access database if based on) to programmers. The DAO interface can also be used to access other relational style data sources.
2. **Remote Data Objects (RDO)** is a higher-level object oriented application interface used to access remote database servers. RDO uses the lower-level DAO and ODBC for direct access to databases. RDO was optimized to deal with server based databases, such as MS SQL Server, Oracle, DB2, and so on.

**4. What are the three basic components of the ODBC architecture?**

* + The basic ODBC architecture is composed of three main components:

1. A high level ODBC API trough which application programs access ODBC functionality.
2. A Driver Manager component that is in charge of managing all database connections.
3. An ODBC Driver component that talks directly to the DBMS (data source).

**5. What steps are required to create an ODBC data source name?**

* + To define a data source you must create a **data source name (DSN)** for the data source. To create a DSN you have to provide:
  1. **An ODBC driver**. You must identify the driver to use to connect to the data source. The ODBC driver is normally provided by the database vendor; although h Microsoft provides several drives to connect to the most common databases. For example, if you are using an Oracle DBMS you will select the Oracle ODBC drive provided by Oracle or, if desired, the Microsoft-provided ODBC Driver for Oracle.
  2. **A DSN name**. This is a unique name by which the data source will be known to ODBC and therefore, to the applications. ODBC offers two types of data sources: User and System. User data sources are only available to the user. System data sources are available to all users, including operating system services.
  3. **ODBC driver parameters**. Most ODBC drivers require some specific parameters in order to establish a connection to the database. For example, if you are using a MS Access database you must point to the location of the MS Access (.mdb) file and, if necessary, provide the user name and password. If you are using a DBMS server, you must provide the server name, the database name, and the user name and password used to connect to the database. Figure 14.3 shows the ODBC screens required to create a System ODBC data source for an Oracle DBMS. Note that some ODBC drivers use the native driver provided by the DBMS vendor.

**7. Explain the OLE-DB model based on its two types of objects.**

* + OLE-DB is composed of a series of COM objects that provide low-level database connectivity for applications. Because OLE-DB is based on the COM object model, the objects contain data and methods (also known as the interface.) The OLE-DB model is better understood when you divide its functionality in two types of objects:
    1. **Consumers** are all those objects (applications or processes) that request and use data. The data consumers request data by invoking the methods exposed by the data provider objects (public interface) and passing the required parameters.
    2. **Providers** are the objects that manage the connection with a data source and provide data to the consumers. Providers are divided in two categories: data providers and service providers.
       1. **Data providers** provide data to other processes. Database vendors create data provider objects that expose the functionality of the underlining data source (relational, object-oriented, text, and so on.)
       2. **Service providers** provide additional functionality to consumers. The service provider is located between the data provider and the consumer: The service provider requests data from the data provider; transforms the data and provides the transformed data to the data consumer. In other words, the service provider acts like a data consumer of the data provider and as a data provider for the data consumer (end-user application). For example, a service provider could offer cursor management services, transaction management services, query processing services, indexing services, and so on.

**8. How does ADO complement OLE-DB?**

* + OLE-DB provided additional capabilities for the applications accessing the data. However, it did not provide support for scripting languages, especially the ones used for web development, such as Active Server Pages (ASP) and ActiveX. To provide such support, Microsoft developed a new object framework called **ActiveX Data Objects** (**ADO**). ADO provides a high level application-oriented interface to interact with OLE-DB, DAO, and RDO. ADO provided a unified interface to access data from any programming language that uses the underlying OLE-DB objects.

**9. What is ADO.NET, and what two new features make it important for application development?**

* + **ADO.NET** is the data access component of Microsoft’s .NET application development framework. Microsoft’s .NET framework is a component-based platform for the development of distributed, heterogeneous, interoperable applications aimed to manipulate any type of data, over any network, and under any operating system and programming language. The .NET framework is beyond the reach of this book. Therefore, this section will only introduce the basic data access component of the .NET architecture, ADO.NET. ADO.Net introduced two new features critical for the development of distributed applications: datasets and XML support.
    1. A DataSet is a disconnected memory-resident representation of the database.
    2. ADO.NET stores all its internal data in XML format.

**10. What is a DataSet, and why is it considered to be disconnected?**

* + A **DataSet** is a disconnected memory-resident representation of the database. That is, the DataSet contains tables, columns, rows, relationships, and constraints. Once the data are read from a data provider, the data are placed on a memory-resident DataSet. The DataSet is then disconnected from the data provider. The data consumer application interacts with the data in the DataSet object to make changes (inserts, updates and deletes) in the dataset. Once the processing is done, the DataSet data are synchronized with the data source and the changes are made permanent.

A DataSet is in fact a simple database with tables, rows and constraints. Even, more important, the DataSet doesn’t require keeping a permanent connection to the data source. The DataAdapter uses the SelectCommand to populate the DataSet from a data source. However, once the DataSet is populated, it is completely independent of the data source – that’s why it’s called “disconnected.”

**16. What is XML, and why is it important?**

* + **Extensible Markup Language** (**XML**) is a meta-language used to represent and manipulate data elements. XML is designed to facilitate the exchange of structured documents such as orders or invoices over the Internet. The World Wide Web Consortium (W3C) published the first XML 1.0 standard definition in 1998. This standard sets the stage for giving XML the real-world appeal of being a true vendor-independent platform. Therefore, it is not surprising that XML is rapidly becoming the data exchange standard for e-commerce applications. XML is important because it provides the semantics that facilitate the sharing, exchange, and manipulation of structured documents over organizational boundaries.