**Chapter 15**

**Database Connectivity and Web Technologies**

**Discussion Focus**

Begin by making sure that the students are familiar with the basic vocabulary. The following two questions are a good place to start. (You may want to examine the contents of Appendix F, “Client/Server Systems.”) There is some irony in the Web development arena … the microcomputer was supposed to liberate the end user from the mainframe computer’s “fat server, thin client” environment. However, the Web has, in effect, brought us back to the old mainframe structure in which most processing is done on the server side, while the client is the source and recipient of data/information requests and returns.

1. **Describe the following: TCP/IP, Router, HTML, HTTP, and URL.**

Transmission Control Protocol/Internet Protocol (TCP/IP) is the basic network protocol that determines the rules used to create and route “packets” of data between computers in the same or different networks. Each computer connected to the Internet has a unique TCP/IP address. The TCP/IP address is divided in two parts used to identify the network and the computer (or host)

Router is a special hardware/software equipment that connects multiple and diverse networks. The router is in charge of delivering packets of data from a local network to a remote network. Routers are the traffic cops of the Internet, monitoring all traffic and moving data from one network to another.

HTML stands for Hyper Text Markup Language is the standard document-formatting language for Web pages. HTML allows documents to be presented in a Web browser in a standard manner.

URL stands for Uniform Resource Locator. An URL identifies the address of a resource on the Internet. The URL is an abbreviation (ideally easily remembered) that uniquely identifies an internet resource, for example [www.amazon.com](http://www.amzon.com), [www.faa.gov](http://www.faa.gov), and [www.mtsu.edu](http://www.mtsu.edu).)

HTTP stands for Hyper Text Transfer Protocol. HTTP is the standard protocol used by the *Web browser* and *Web server* to communicate—that is, to send requests and replies between servers and browsers. HTTP uses TCP/IP to transmit the data between computers.

1. **Describe the client/server model for application processing.**

**Client/server** is a term used to describe a computing model for the development of computerized systems. This model is based on the distribution of functions between two types of independent and autonomous processes: servers and clients. A **client** is any process that requests specific services from server processes. A **server** is a process that provides requested services for clients. Client and server processes can reside in the same computer or in different computers connected by a network.

The client/server model makes possible the division of the application processing tasks into three main components: presentation logic, processing logic, and data storage.

* The *presentation logic* formats and presents data in output devices, such as the screen, and manages the end-user input. The application uses presentation logic to manage the graphical user interface at the client end.
* The *processing logic* component refers to the application code that performs data validation, error checking, and business logic processing. The processing logic component represents the business rules. For example, the processing logic “knows” that a sales transaction generates an invoice record, an inventory update, and a customer’s account receivable update. The processing logic performs several functions, including enforcement of business rules, managing information flows within the business, and mapping the real-world business transactions to the actual computer database.
  + - The *data storage* component deals with the actual data storage and retrieval from permanent storage devices. For example, the data manipulation logic is used to access the data in a database and to enforce data integrity.

Although there is no methodology to dictate the precise distribution of the logic components among clients and servers, the client/server architectural principles of process distribution (autonomy, resource maximization, scalability, and interoperability) and hardware and software independence facilitate the creation of distributed applications running across multiple servers. Those applications provide services that communicate with each other in order to carry out specific function, therefore the term multi-tier applications. So, where should the services be placed? With the probable exception of the presentation logic, which should go on the client side, each of the remaining service components may be placed on the server side, thus becoming a service for many clients.

**Answers to Review Questions**

1. **Give some example 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. Although there are many different ways to achieve database connectivity, this section will cover only the following interfaces: native SQL connectivity (vendor provided), Microsoft’s Open Database Connectivity (ODBC), Data Access Objects (DAO) and Remote Data Objects (RDO), Microsoft’s Object Linking and Embedding - Databases (OLE-DB) and Microsoft’s ActiveX Data Objects (ADO.NET) and Java Database Connectivity (JDBC).

1. **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:

* **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.
* **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.

1. **What is the difference between DAO and RDO?**

DAO is uses the MS Jet engine to access file-based relational databases such as MS Access, MS FoxPro and Dbase. In contrast, RDO allows to access relational database servers such as SQL Server, DB2, and Oracle. RDO uses DAO and ODBC to access remote database server data.

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

The basic ODBC architecture is composed of three main components:

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

1. **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:

* 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.
* 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.
* *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 15.3 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.

1. **What is OLE-DB used for, and how does it differ from ODBC?**

Although ODBC, DAO, and RDO were widely used, they did not provide support for non-relational data. To answer the need for non-relational data access and to simplify data connectivity, Microsoft developed **Object Linking and Embedding for Database** (**OLE-DB**). Based on Microsoft’s Component Object Model (COM), OLE-DB is a database middleware that was developed to add object-oriented functionality for access to relational and non-relational data. OLE-DB was the first piece of Microsoft’s strategy to provide a unified object-oriented framework for the development of next-generation applications.

1. **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:

* ***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.
* ***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*.
  + **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.)
  + **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.

1. **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. Figure 15.5 – borrowed from the text and reproduced here for your convenience -- illustrates the ADO/OLE-DB architecture and how it interacts with ODBC and native connectivity options.

**Figure 15.5 OLE-DB ARCHITECTURE**



1. **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.

* A **DataSet** is a *disconnected memory-resident representation of the database*.
* ADO.NET stores all its internal data in XML format.

1. **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.”

1. **What are Web server interfaces used for? Give some examples.**

Web server interfaces are used to extend the functionality of the web server to provide more services. If a Web server is to successfully communicate with other external programs to provide a service, both programs must use a standard way to exchange messages and respond to requests. A Web server interface defines how a Web server communicates with external programs. Currently there are two well-defined Web server interfaces:

* Common Gateway Interface (CGI)
* Application Programming Interface (API)

Web server interfaces can be used to extend the services of a web server and provide support for access to external databases, fax services, telephony services, directory services, etc.

1. **Search the Internet for Web application servers. Choose one and prepare a short presentation for your class.**

You are encouraged to use any web search engine to list multiple vendors. Examples of such vendors are: Oracle Application Server, IBM WebSphere, Sun Java, Microsoft, JBOSS, etc. We encourage the student to visit the web pages of the products and compare features of at least two products. Some of the many other Web application servers, as of this writing, include Oracle Application Server by Oracle Corp., WebLogic by BEA Systems, NetDynamics by Sun Microsystems, NetObjects’ Fusion, Microsoft’s Visual Studio.NET, and WebObjects by Apple.

1. **What does this statement mean: The Web is a stateless system? What implications does a stateless system have for database applications developers?**

Simply put, the label **stateless system** indicates that, at any given time, a Web server does not know the status of any of the clients communicating with it. That is, there is no open communications line between the server and each client accessing it -- that, of course, is impractical in a *worldwide* Web! Instead, client and server computers interact in very short “conversations” that follow the request-reply model. For example, the browser is only concerned with the *current* page, so there is no way for the second page to know what was done in the first page. The only time the client and server computers communicate is when the client requests a page—when the user clicks a link—and the server sends the requested page to the client. Once the client receives the page and its components, the client/server communication is ended. Therefore, although you may be browsing a page and *think* that the communication is open, you are actually just browsing the HTML document stored in the local cache (temporary directory) of the client browser. The server does not have any idea what the end user is doing with the document, what data is entered in a form, what option is selected, and so on. In the Web, if we want to act on a client’s selection, we need to jump to a new page (go back to the Web server), therefore losing track of whatever was done before!

Not knowing what was done before, or what a client selected before it got to this page, makes adding business logic to the Web cumbersome. For example, suppose that you need to write a program that performs the following steps: display a data entry screen, capture data, validate data, and save data. This entire sequence can be completed in a single COBOL program because COBOL uses a working storage section that holds in memory all variables used in the program. Now imagine the same COBOL program -- but *each* section (PERFORM statement) is now a separate program! That is precisely how the Web works. In short, the Web’s stateless nature means that extensive processing required by a program’s execution cannot be done directly in a single Web page; the client browser’s processing ability is limited by the lack of processing ability and the lack of a working storage area to hold variables used by all pages in a Web site.

The browser does not have computational abilities beyond formatting output text and accepting form field inputs. Even when the browser accepts form field data, there is no way to perform immediate data entry validation. Therefore, to perform such crucial processing in the client, the Web defers to other Web programming languages such as Java, JavaScript, and VBScript.

1. **What is a Web application server, and how does it work from a database perspective?**

Web application server extends the functionality of a web server and provides features such as:

* An integrated development environment with session management and support for persistent application variables.
* Security and authentication of users through user IDs and passwords.
* Computational languages to represent and store business logic in the application server.
* Automatic generation of HTML pages integrated with Java, JavaScript, VBScript, ASP, and so on.
* Performance and fault-tolerant features.
* Database access with transaction management capabilities.
* Access to multiple services, such as file transfers (FTP), database connectivity, electronic mail, and directory services.

The web application server interfaces with the database connectivity standards to access databases using any of the supported API. So, a web page will be processed by the web application server, the application server will connect to the database using the ADO, OLE-DB or ODBC standard (or any other standard supported by the application server).

1. **What are scripts, and what is their function? (Think in terms of database applications development!)**

A **script** is a series of instructions executed in *interpreter* mode. The script is a plain text file that is not compiled like COBOL, C++, or Java. Scripts are normally used in web application development environments. For instance, ColdFusion scripts contain the code that is required to connect, query, and update a database from a Web front end.

1. **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.

1. **What are document type definition (DTD) documents and what do they do?**

Companies that exchange data using XML must have a way to understand and validate each other’s tags. One way to accomplish that task is through the use of Document Type Definitions. A **Document Type Definition** (**DTD**) is a file with a .dtd extension that describes XML elements—in effect, a DTD file provides the composition of the database’s logical model, and defines the syntax rules or valid tags for each type of XML document. (The DTD component is very similar to having a public data dictionary for business data.)

1. **What are XML schema definition (XSD) documents and what do they do?**

An **XML Schema Definition** (**XSD**) document is an advanced data definition language that is used to describe the structure (elements, data types, relationship types, ranges, and default values) of XML data documents. Unlike a DTD document, which uses a unique syntax, an XML Schema Definition (XSD) file uses a syntax that resembles an XML document. One of the main advantages of an XML schema is that it more closely maps to database terminology and features. For example, an XML schema will be able to define common database types, such as date, integer or decimal, minimum and maximum values, list of valid values, and required elements. Using the XML schema, a company would be able to validate the data for values that may be out of range, incorrect dates, valid values, and so on. For example, a university application must be able to specify that a GPA value must be between zero and 4.0 and it must be able to detect an invalid birth date such as “14/13/1987.” (There is no 14th month.) Many vendors are rapidly adopting this new standard and are supplying tools to translate DTD documents into XML Schema Definition (XSD) documents. It is widely expected that XML schemas will replace DTD as the method to describe XML data.

1. **What is a JDBC, and what is it used for?**

JDBC (Java Database Connectivity) is discussed in detail in Section 15-1e.

Java is an object-oriented programming language developed by Sun Microsystems that runs on top of Web browser software. Java is one of the most common programming languages for Web development. Sun Microsystems created Java as a “write once, run anywhere” environment. That means that a programmer can write a Java application once and then without any modification, run the application in multiple environments (Microsoft Windows, Apple OS X, IBM AIX, etc.). The cross-platform capabilities of Java are based on its portable architecture. Java code is normally stored in pre-processed chunks known as applets that run on a virtual machine environment in the host operating system. This environment has well-defined boundaries and all interactivity with the host operating system is closely monitored. Sun provides Java runtime environments for most operating systems (from computers to hand-held devices to TV set-top boxes.) Another advantage of using Java is its “on-demand” architecture. When a Java application loads, it can dynamically download all its modules or required components via the Internet.

When Java applications want to access data outside the Java runtime environment, they use pre-defined application programming interfaces. **Java Database Connectivity (JDBC)** is an application programming interface that allows a Java program to interact with a wide range of data sources (relational databases, tabular data sources, spreadsheets, and text files). JDBC allows a Java program to establish a connection with a data source, prepare and send the SQL code to the database server, and process the result set.

One of the main advantages of JDBC is that it allows a company to leverage its existing investment in technology and personnel training. JDBC allows programmers to use their SQL skills to manipulate the data in the company’s databases. As a matter of fact, JDBC allows direct access to a database server or access via database middleware. Furthermore, JDBC provides a way to connect to databases through an ODBC driver. (Figure 15.7 in the text illustrates the basic JDBC architecture and the various database access styles.)

The database access architecture in JDBC is very similar to the ODBC/OLE/ADO.NET architecture. All database access middleware shares similar components and functionality. One advantage of JDBC over other middleware is that it requires no configuration on the client side. The JDBC driver is automatically downloaded and installed as part of the Java applet download. Because Java is a Web-based technology, applications can connect to a database directly using a simple URL. Once the URL is invoked, the Java architecture comes into place, the necessary applets are downloaded to the client (including the JDBC database driver and all configuration information), and then the applets are executed securely in the client’s runtime environment.

Every day, more and more companies are investing resources in developing and expanding their Web presence and finding ways to do more business on the Internet. Such business will generate increasing amounts of data that will be stored in databases. Java and the .NET framework are part of the trend toward increasing reliance on the Internet as a critical business resource. In fact, it has been said that the Internet will become the development platform of the future. In the next section you will learn more about Internet databases and how they are used.

1. **What is cloud computing, and why is it a “game changer”?**

According to the National Institute of Standards and Technology (NIST), cloud computing is “a computing model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computer resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” The term cloud services is used in this book to refer to the services provided by cloud computing. Cloud services allow any organization to quickly and economically add information technology services such as applications, storage, servers, processing power, databases, and infrastructure to its IT portfolio (see Figure 15.21).

Cloud computing is important for database technologies because it has the potential to become a “game changer.” Cloud computing eliminates financial and technological barriers so organizations can leverage database technologies in their business processes with minimal effort and cost. In fact, cloud services have the potential to turn basic IT services into “commodity” services such as electricity, gas, and water, and to enable a revolution that could change not only the way that companies do business, but the IT business itself. As Nicholas Carr put it so vividly: “Cloud computing is for IT what the invention of the power grid was for electricity.”

For example, imagine that the chief technology officer of a nonprofit organization wants to add e-mail services to the IT portfolio. A few years ago, this proposition would have implied building the e-mail system’s infrastructure from the ground up, including hardware, software, setup, configuration, operation, and maintenance. However, in today’s cloud computing era, you can use Google Apps for Business or Microsoft Exchange Online and get a scalable, flexible, and more reliable e-mail solution for a fraction of the cost. The best part is that you do not have to worry about the daily chores of managing and maintaining the IT infrastructure, such as OS updates, patches, security, fault tolerance, and recovery. What used to take months or years to implement can now be done in a matter of minutes.

1. **Name and contrast the types of cloud computing implementation.**

Section 15-4a describes this item in detail. There are basically three cloud computing implementation types (based on who the target customers are):

* Public cloud. This type of cloud infrastructure is built by a third-party organization to sell cloud services to the general public. The public cloud is the most common type of cloud implementation; examples include Amazon Web Services (AWS), Google Application Engine, and Microsoft Azure. In this model, cloud consumers share resources with other consumers transparently. The public cloud infrastructure is managed exclusively by the third-party provider.
* Private cloud. This type of internal cloud is built by an organization for the sole purpose of servicing its own needs. Private clouds are often used by large, geographically dispersed organizations to add agility and flexibility to internal IT services. The cloud infrastructure could be managed by internal IT staff or an external third party.
* Community cloud. This type of cloud is built by and for a specific group of organizations that share a common trade, such as agencies of the federal government, the military, or higher education. The cloud infrastructure could be managed by internal IT staff or an external third party.

1. **Name and describe the most prevalent characteristics of cloud computing services.**

Section 15-4b describes the basic characteristics of cloud computing services. In summary, the characteristics are:

* Ubiquitous access via Internet.
* Shared infrastructure.
* Lower costs and variable pricing.
* Flexible and scalable services.
* Dynamic provisioning.
* Service orientation.
* Managed operations.

1. **Using the Internet, search for providers of cloud services. Then, classify the types of services they provide (SaaS, PaaS, and IaaS).**

A starting point will be the examples shown in Figure 15.22. Further examples are:

* DropBox.com – a cloud service storage provider (IaaS)
* Carbonite.com – provide online backup of data (SaaS)
* iCloud.com (Apple) – provides storage and synchronization of Apple device data (contacts, music, apps, photos, documents, backups (IaaS and SaaS)
* GoodData.com – Business intelligence platform (PaaS)
* Heroku.com – Ruby web programing environment service (PaaS)

1. **Summarize the main advantages and disadvantages of cloud computing services.**

Table 15.4 summarizes the main advantages and disadvantages of cloud computing services.

1. **Define SQL data services and list their advantages.**

SQL data services refer to Internet-based data management services that provide access to hosted relational data management using standard protocols and common programming interfaces. The advantages of SQL data services include:

* High reliability and scalability of relational database capabilities at a low cost
* High level of failure tolerance
* Dynamic and automatic load balancing
* Automated data backup and recovery
* Dynamic creation and allocation of database processes and storage.

**Problem Solutions**

|  |
| --- |
| **ONLINE CONTENT**  **The databases used in the Problems for this chapter can be found at *www.cengagebrain.com*.** |

PROBLEMS

In the following exercises, you set up database connectivity using MS Excel.

**NOTE:** Although the precise steps to setup data connectivity vary slightly according to the version of Excel and operating system platform you are using, in general terms, the steps are outlined as indicated in the sections below.

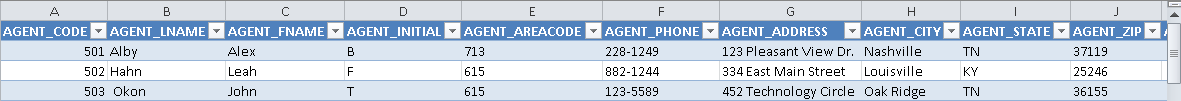
1. **Use MS Excel to connect to the Ch02\_InsureCo MS Access database using ODBC, and retrieve all of the AGENTs.**

To perform this task, complete the following steps:

* From Excel, select **Data, From Other Sources, From Microsoft Query** options to retrieve data from an ODBC data source.
* Select the **MS Access Database\*** option and click **OK**.
* Select the **Database** file location and click **OK**.
* Select the table and columns to use in the query (select **all columns**) and click **Next**.
* On the Query Wizard – **Filter Data** click **Next.**
* On the Query Wizard – **Sort Order** click **Next.**
* Select **Return Data to Microsoft Office Excel**.
* Position the cursor where you want the data to be placed on your spreadsheet and click **OK**.

The solution is shown in Figure P15.1.

**Figure P15.1 Solution to Problem 1 – Retrieve all AGENTs**



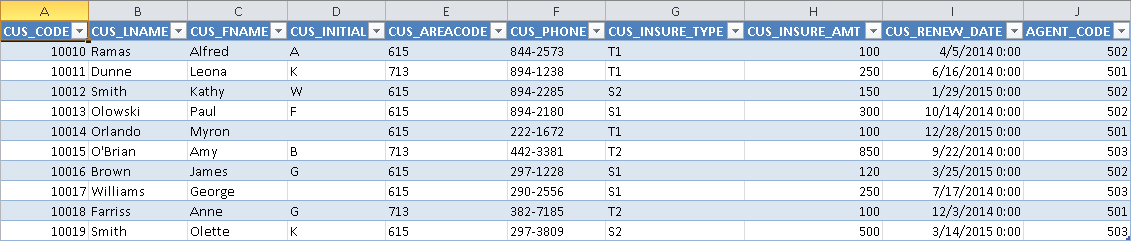
1. **Use MS Excel to connect to the Ch02\_InsureCo MS Access database using ODBC, and retrieve all of the CUSTOMERs.**

To perform this task, complete the following steps:

* From Excel, select **Data**, **From Other Sources, From Microsoft Query** options to retrieve data from an ODBC data source.
* Select the **MS Access Database\*** option and click **OK**.
* Select the **Database** file location and click **OK**.
* Select the table and columns to use in the query (select **all columns**) and click **Next**.
* On the Query Wizard – **Filter Data** click **Next.**
* On the Query Wizard – **Sort Order** click **Next.**
* Select **Return Data to Microsoft Office Excel**.
* Position the cursor where you want the data to be placed on your spreadsheet and click **OK**.

The solution is shown in Figure P15.2.

**Figure P15.2 Solution to Problem 2 – Retrieve all CUSTOMERs**



1. **Use MS Excel to connect to the Ch02\_InsureCo MS Access database using ODBC, and retrieve the customers whose AGENT\_CODE is equal to 503.**

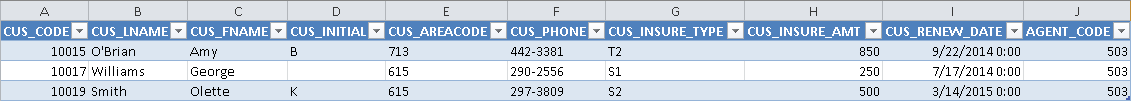
To perform this task, complete the following steps:

* From Excel, select **Data, From Other Sources, From Microsoft Query** options to retrieve data from an ODBC data source.
* Select the **MS Access Database\*** option and click **OK**.
* Select the **Database file location** and click **OK**.
* Select the table and columns to use in the query (select **all columns**) and click **Next**.
* On the Query Wizard – **Filter Data**, select the **AGENT\_CODE** column, select “**equals**” from the left drop down box, then select “**503**” from the right drop down box, and then click **Next**.
* On the Query Wizard – **Sort Order** click **Next**.
* Select **Return Data to Microsoft Office Excel**.
* Position the cursor where you want the data to be placed on your spreadsheet and click **OK**.

The results are shown in Figure P15.3.

**Figure P15.3 Solution to Problem 3 –**

**Retrieve all CUSTOMERs with AGENT\_CODE=503**



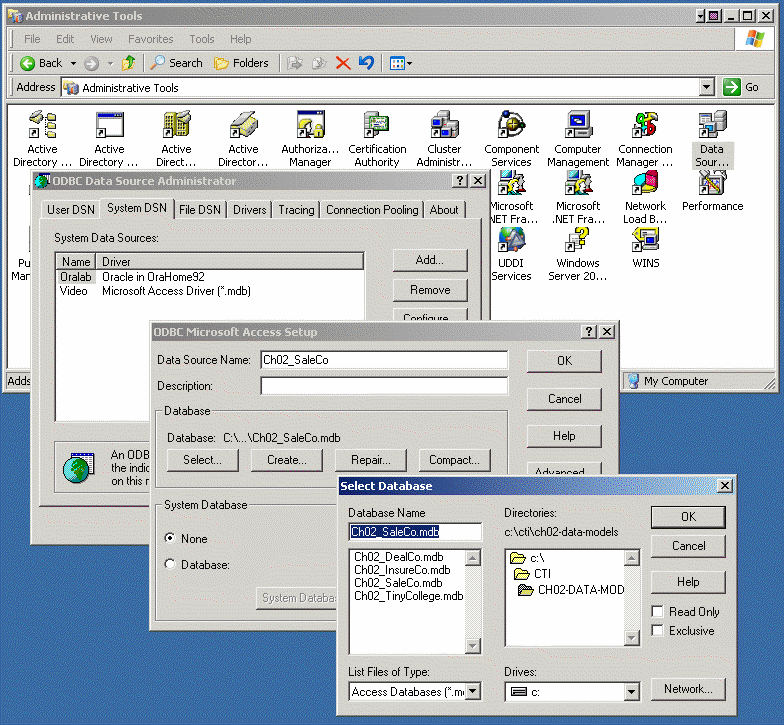
1. **Create a System DSN ODBC connection called Ch02\_SaleCo using the Administrative Tools section of the Windows Control Panel.**

To create the DSN, complete the following steps:

* Using Windows XP, open the **Control Panel**, open **Administrative Tools**, open **Data Sources (ODBC)**.
* Click on the **System DSN** tab, click on **Add**, select the **Microsoft Access Drive (\*.mdb)** driver and click on **Finish**.
* On the ODBC Microsoft Access Setup window, enter the **Ch02\_SaleCo** on the **Data Source Name** field.
* Under **Database**, click on the **Select** button, browse to the location of the MS Access file and click **OK** twice.
* The new system DSN now appears in the list of system data sources.

The results are shown in Figure P15.4.

**Figure P15.4 Solution to Problem 4 – Create Ch02\_SaleCo System DSN**



1. **Use MS Excel to list all of the invoice lines for Invoice 1003 using the Ch02\_SaleCo System DSN.**

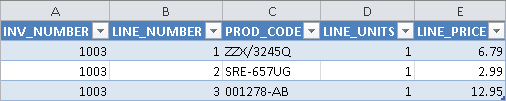
To perform this task, complete the following steps:

* From Excel, select **Data**, **Import External Data** and **New Database Query** options to retrieve data from an ODBC data source.
* Select the **Ch02\_SaleCo** data source and click OK.
* Select the **LINE** table, then select **all columns**, and click **Next**.
* On the Query Wizard – **Filter Data**, select the **INV\_NUMBER** column, select “**equals**” from the left drop down box, then select “**1003**” from the right drop down box, and then click **Next**.
* On the Query Wizard – **Sort Order** click **Next**.
* Select **Return Data to Microsoft Office Excel**.
* Position the cursor where you want the data to be placed on your spreadsheet and click **OK**.

The results are shown in Figure P15.5.

**Figure P15.5 Solution to Problem 5 –**

**Retrieve all invoice LINEs with INV\_NUMBER=1003**



1. **Create a System DSN ODBC connection called Ch02\_TinyCollege using the Administrative Tools section of the Windows Control Panel.**

To perform this task, complete the following steps:

* Using Windows XP, open the **Control Panel**, open **Administrative Tools**, **open Data Sources (ODBC)**.
* Click on the **System DSN** tab, click on **Add**, select the **Microsoft Access Drive (\*.mdb)** driver and click on **Finish**.
* On the ODBC Microsoft Access Setup window, enter the **Ch02\_TinyCollege** on the **Data Source Name** field.
* Under **Database**, click on the **Select** button, browse to the location of the MS Access file and click **OK** twice.
* The new system DSN now appears in the list of system data sources.

1. **Use MS Excel to list all classes taught in room KLR200 using the Ch02\_TinyCollege System DSN.**

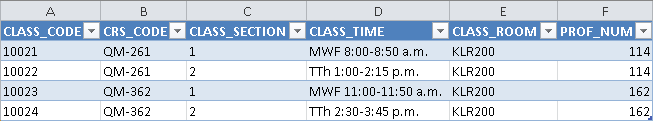
To perform this task, complete the following steps:

* From Excel, select **Data**, **Import External Data** and **New Database Query** options to retrieve data from an ODBC data source.
* Select the **Ch02\_TinyCollege** data source and click **OK**.
* Select the **CLASS** table, select **all columns** and click **Next**.
* On the Query Wizard – **Filter Data**, select the **CLASS\_ROOM** column, select “**equals**” from the left drop down box, then select “**KLR200**” from the right drop down box, and then click **Next**.
* On the Query Wizard – **Sort Order** click **Next**.
* Select **Return Data to Microsoft Office Excel**.
* Position the cursor where you want the data to be placed on your spreadsheet and click **OK**.

The results of these actions are shown in Figure P15.7.

**Figure P15.7 Solution to Problem 7 –**

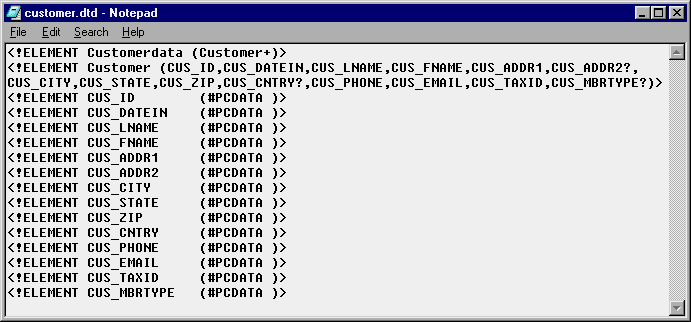
**Retrieve all classes taught in room KLR200**



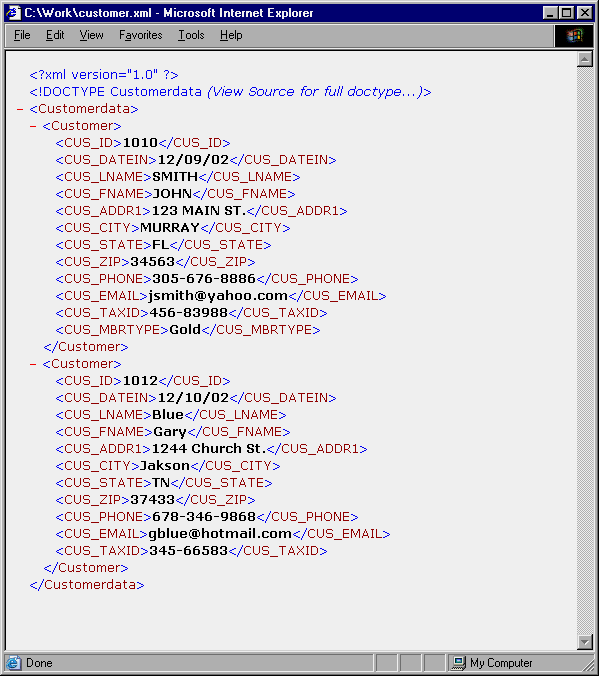
1. **Create a sample XML document and DTD for the exchange of customer data.**

The solutions are shown in Figures P15.8a and P15.8b.

**Figure P15.8a Customer DTD Solution**



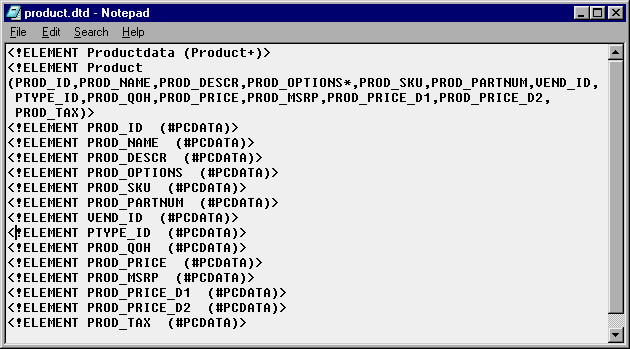
**Figure P15.8b Customer XML Solution**



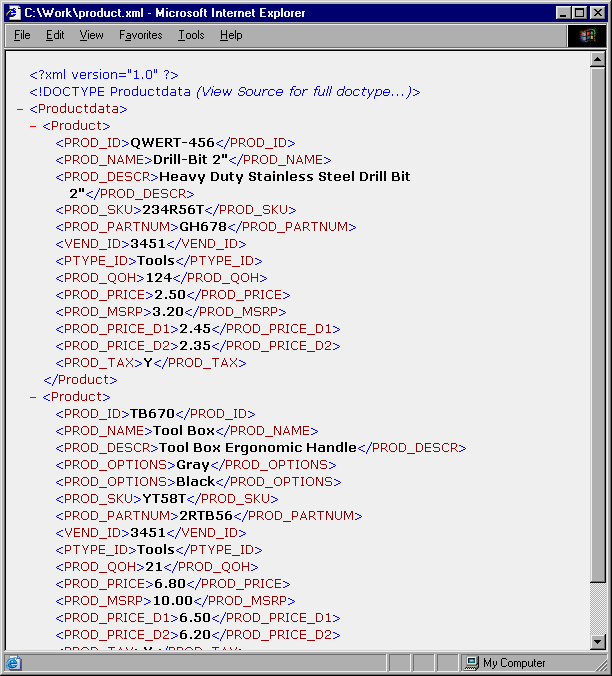
1. **Create a sample XML document and DTD for the exchange of product and pricing data.**

The solutions are shown in Figures P15.9a and P15.9b.

**Figure P15.9a Product DTD Solution**



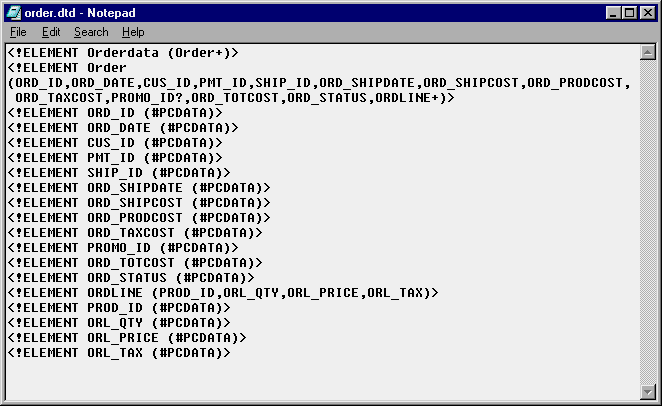
**Figure P15.9b Product XML Solution**



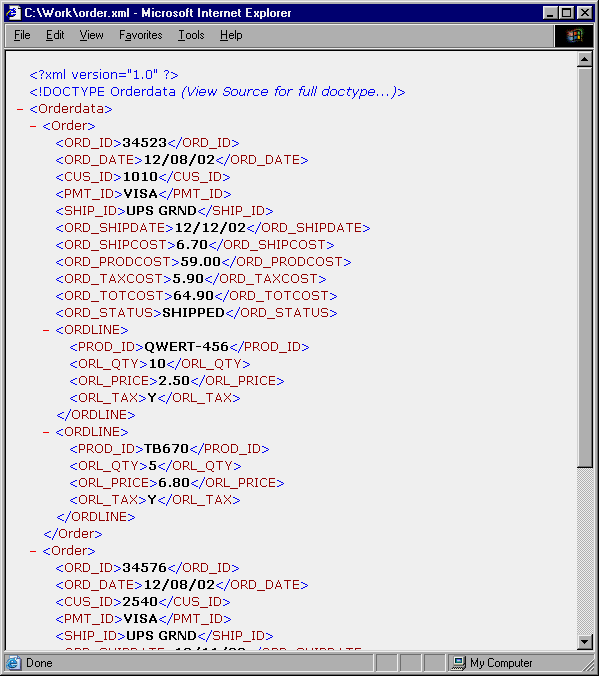
1. **Create a sample XML document and DTD for the exchange of order data.**

The solutions are shown in Figures P15.10a and P15.10b.

**Figure P15.10a Order DTD Solution**



**Figure P15.10b Order XML Solution**



1. **Create a sample XML document and DTD for the exchange of student transcript data. Use your college transcript as a sample.**

The solution to Problem 11 will follow the same format as the previous solutions. However, because Problem 11 requires the students to do some research regarding the information that goes in the transcript data, we have not included a specific solution here. Encourage the student to use his/her creativity and analytical skills to research and create a simple XML file containing the data that is customary on your university.

Not all fields in the Student transcript must be included in this exercise. Allow the students to represent just the most important fields.