Database Systems: Design, Implementation, and Management

Ninth Edition

Chapter 11

Database Connectivity and Web Technologies

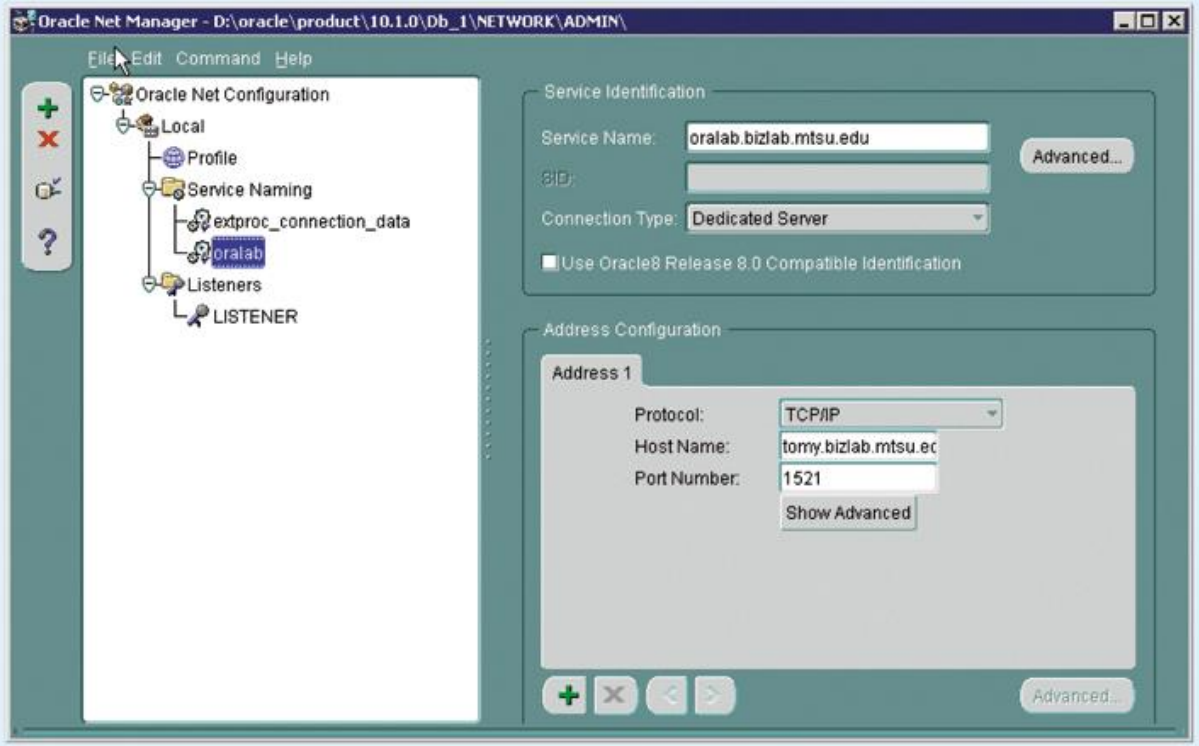
Database Connectivity

- Mechanisms by which application programs connect and communicate with data sources
  - Also known as **database middleware**
- Data repository:
  - Also known as a data source
  - Represents the data management application
    - Used to store data generated by an application program
- ODBC, OLE-DB, ADO.NET: the backbone of MS **Universal Data Access (UDA)** architecture

Native SQL Connectivity

- Connection interface provided by database vendors
  - Unique to each vendor
- Example: Oracle RDBMS
  - Must install and configure Oracle's SQL*Net interface in client computer
- Interfaces optimized for particular vendor's DBMS
  - Maintenance is a burden for the programmer

FIGURE 14.1 ORACLE native connectivity



ODBC, DAO, and RDO

- **Open Database Connectivity (ODBC)**
  - Microsoft's implementation of a superset of SQL Access Group **Call Level Interface (CLI)**
  - Widely supported database connectivity interface
  - Any Windows application can access relational data sources
  - Uses SQL via standard **application programming interface (API)**

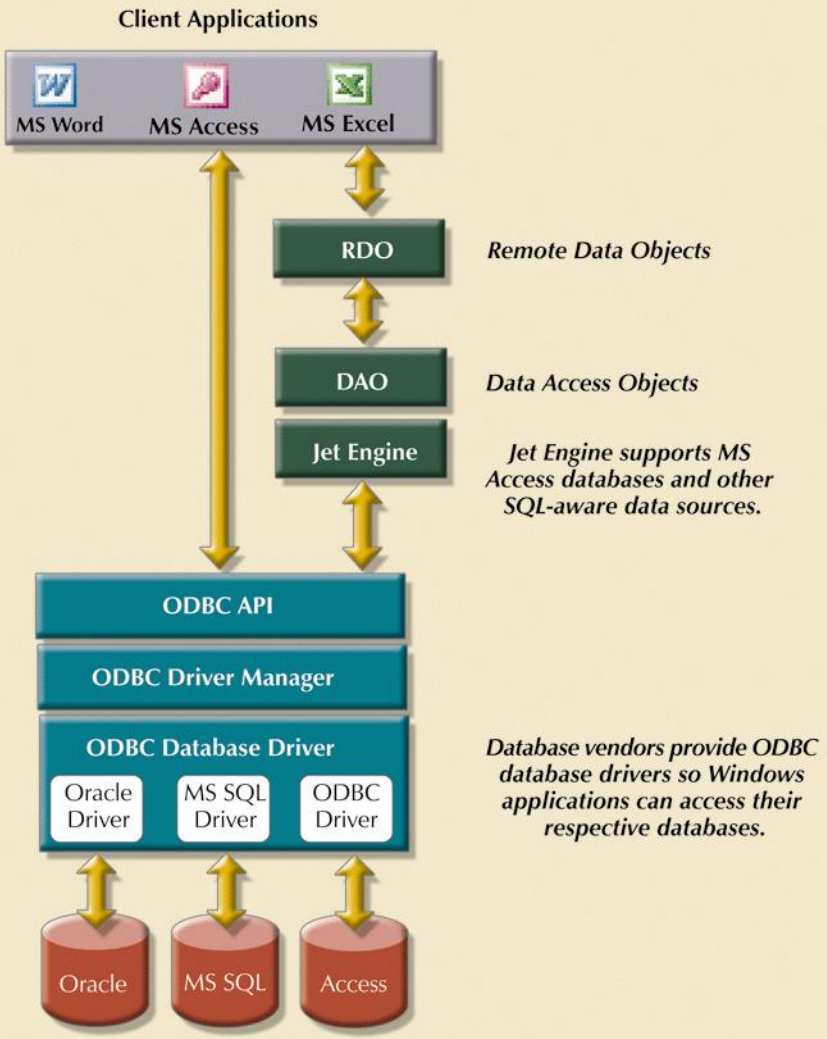
ODBC, DAO, and RDO (cont'd.)

- **Data Access Objects (DAO)**
  - Object-oriented API
    - Accesses MS Access, MS FoxPro, and dBase databases from Visual Basic programs
  - Provided an optimized interface that exposed functionality of Jet data engine to programmers
  - DAO interface can also be used to access other relational style data sources

ODBC, DAO, and RDO (cont'd.)

- **Remote Data Objects (RDO)**
  - Higher-level object-oriented application interface used to access remote database servers
  - Uses lower-level DAO and ODBC for direct access to databases
  - Optimized to deal with server-based databases, such as MS SQL Server, Oracle, and DB2
- Implemented as shared code dynamically linked to Windows via **dynamic-link libraries**

FIGURE 14.2 Using ODBC, DAO, and RDO to access databases



ODBC, DAO, and RDO (cont'd.)

- Basic ODBC architecture has three main components:
  - High-level ODBC API through which application programs access ODBC functionality
  - Driver manager that is in charge of managing all database connections
  - ODBC driver that communicates directly to DBMS

FIGURE 14.3 Configuring an Oracle ODBC data source

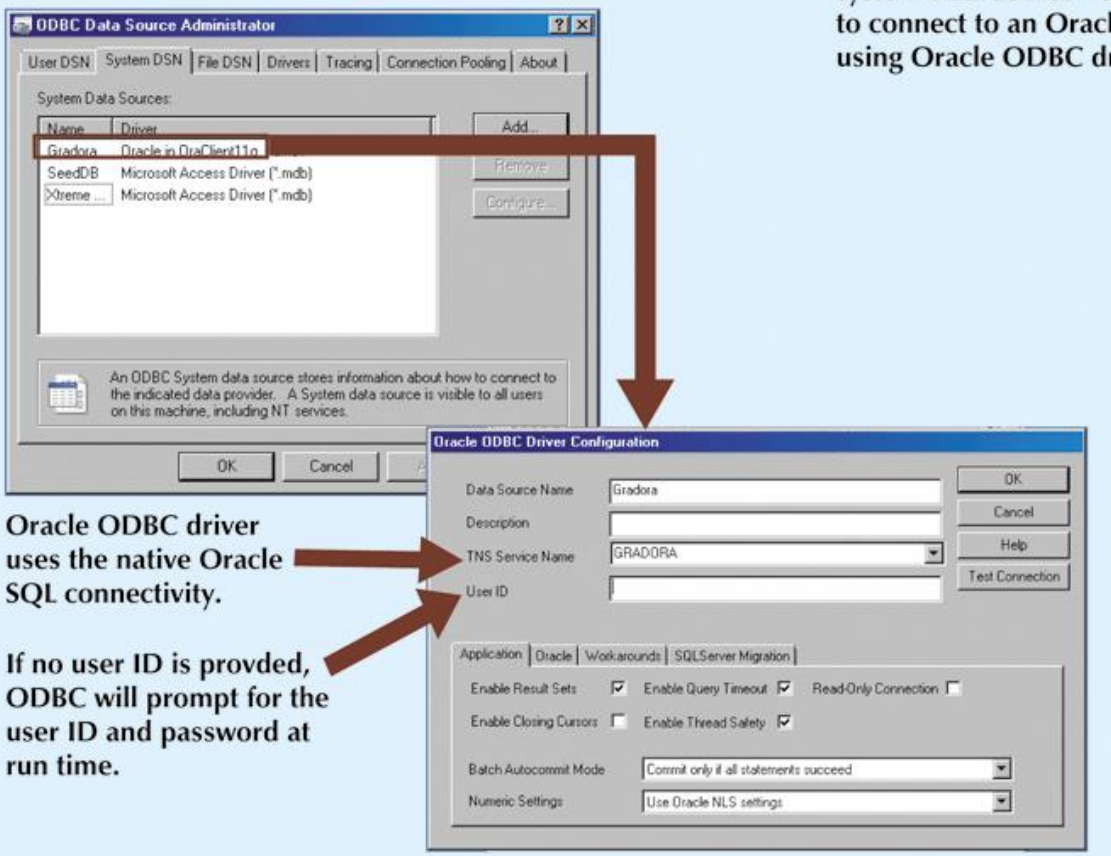
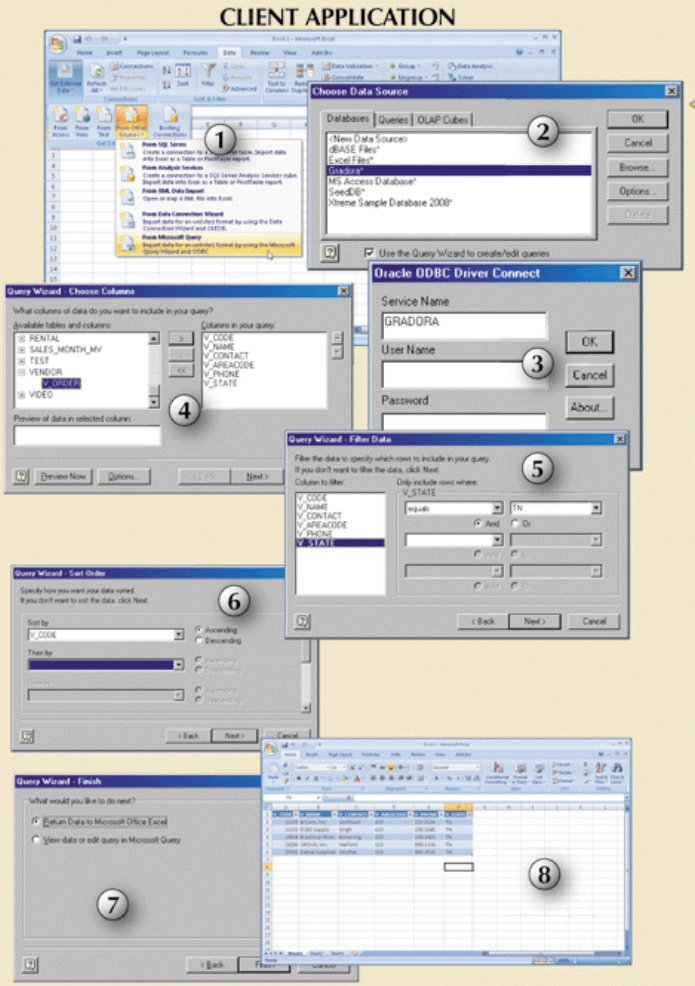
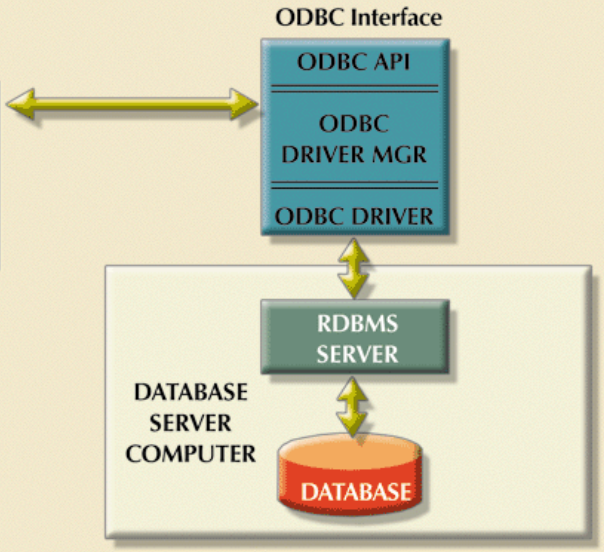


FIGURE 14.4 MS Excel uses ODBC to connect to an Oracle database





1. From Excel, select Get External Data, From Other Sources and From Microsoft Query options to retrieve data from an Oracle RDBMS.
2. Select the Gradora ODBC data source.
3. Enter the authentication parameters. ODBC uses the connection parameters to connect to the data source.
4. Select the table and columns to use in the query.
5. Select filtering options to restrict the rows returned.
6. Select sorting options to order the rows.
7. Select Return Data to Microsoft Office Excel.
8. Excel uses the ODBC API to pass the SQL request down to the database. Oracle executes the request and generates a result set. Excel issues calls to the ODBC API to retrieve the result set and populate the spreadsheet.

OLE-DB

- **Object Linking and Embedding for Database**
- Database middleware that adds object-oriented functionality for access to data
- Series of COM objects provides low-level database connectivity for applications
- Functionality divided into two types of objects:
  - Consumers
  - Providers

OLE-DB (cont'd.)

- OLE-DB did not provide support for scripting languages
- ActiveX Data Objects (ADO) provides high-level application-oriented interface to interact with OLE-DB, DAO, and RDO
- ADO provides unified interface to access data from any programming language that uses the underlying OLE-DB objects

TABLE 14.1 Sample OLE-DB Classes and Interfaces

OBJECT CLASS	USAGE	SAMPLE INTERFACES
Session	Used to create an OLE-DB session between a data consumer application and a data provider.	IGetDataSource ISessionProperties
Command	Used to process commands to manipulate a data provider's data. Generally, the command object will create RowSet objects to hold the data returned by a data provider.	ICommandPrepare ICommandProperties
RowSet	Used to hold the result set returned by a relational-style database or a database that supports SQL. Represents a collection of rows in a tabular format.	IRowsetInfo IRowsetFind IRowsetScroll

FIGURE 14.5 OLE-DB architecture

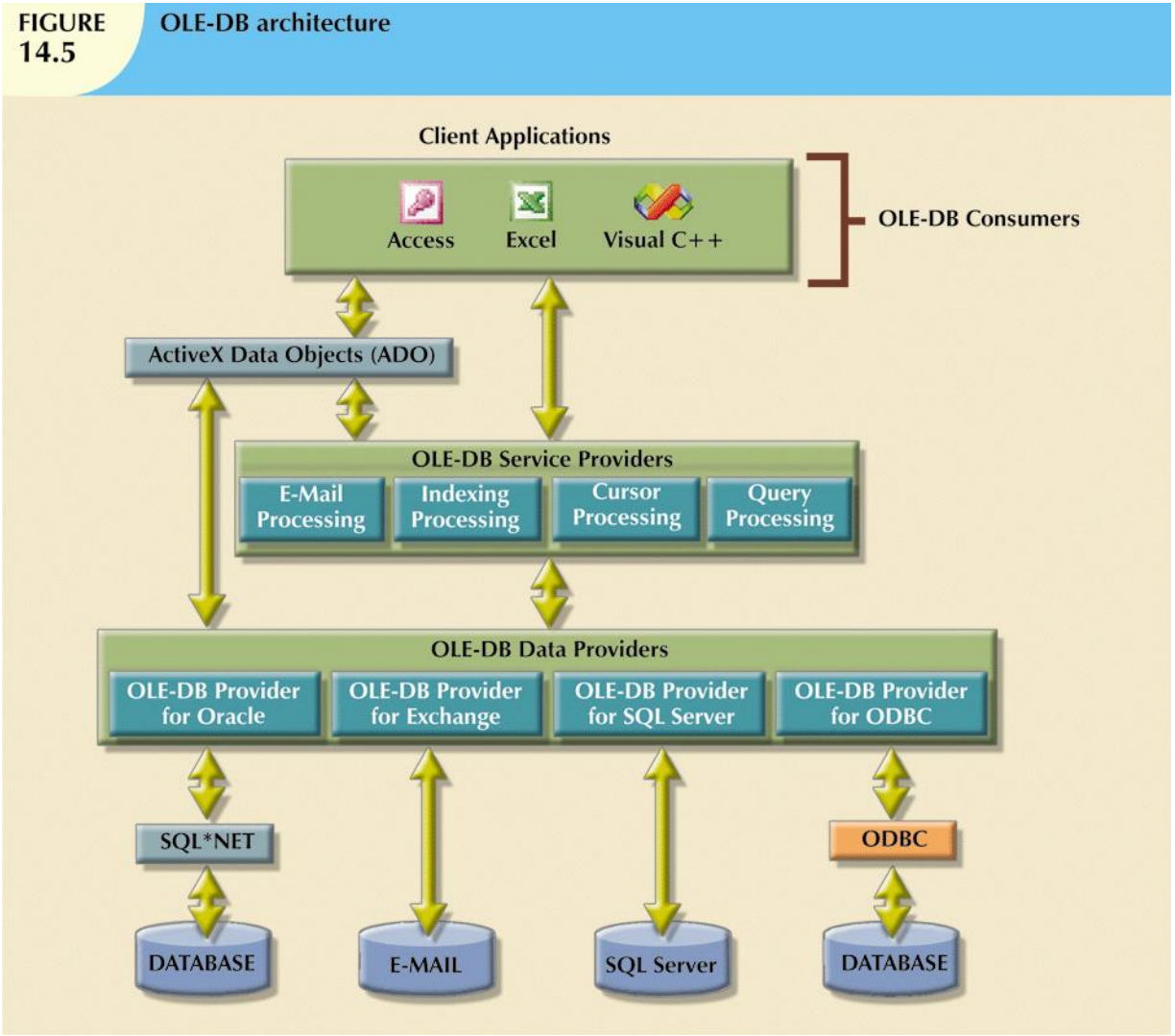


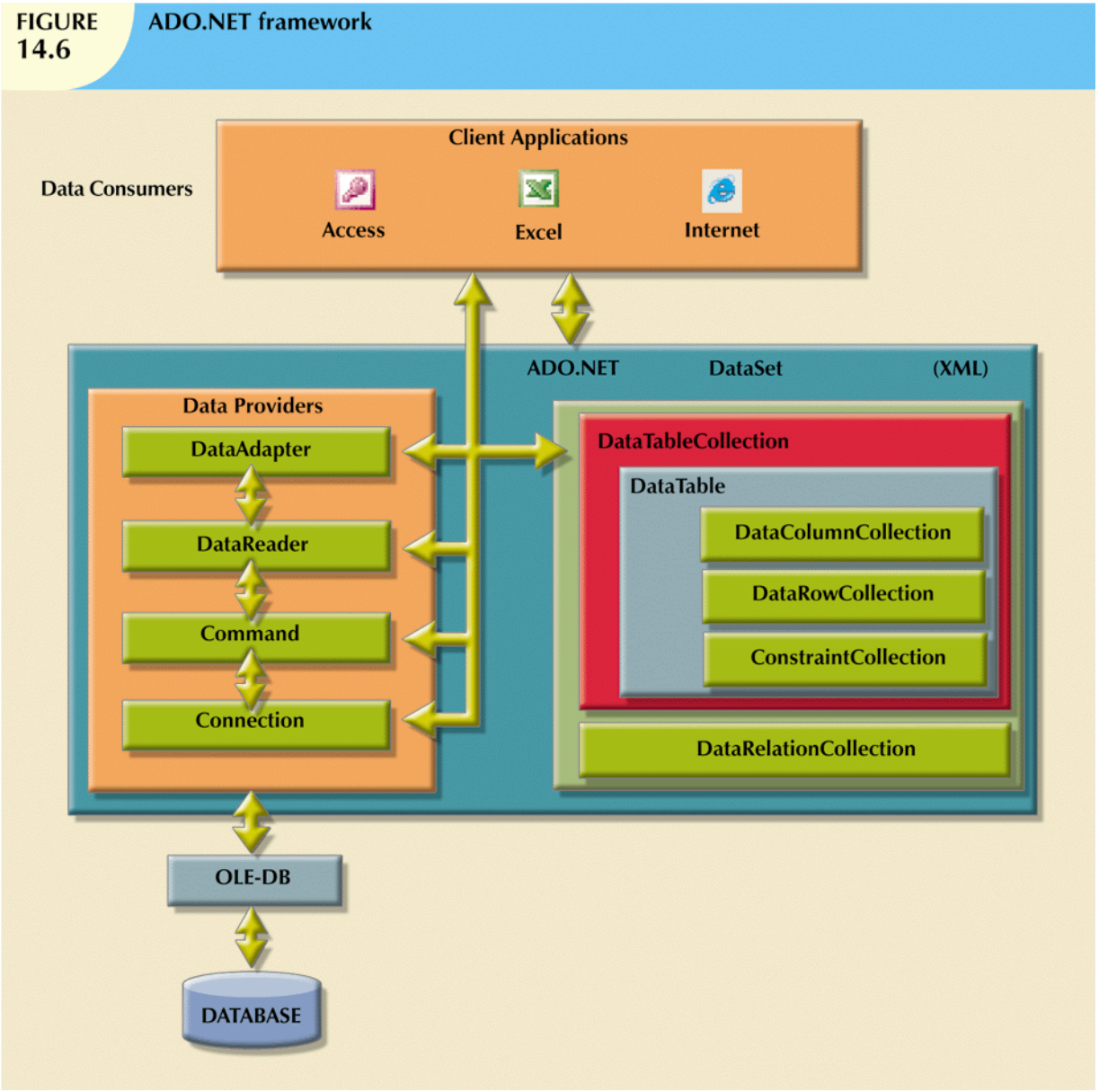
TABLE 14.2 Sample ADO Objects

OBJECT CLASS	USAGE
Connection	Used to set up and establish a connection with a data source. ADO will connect to any OLE-DB data source. The data source can be of any type.
Command	Used to execute commands against a specific connection (data source).
Recordset	Contains the data generated by the execution of a command. It will also contain any new data to be written to the data source. The Recordset can be disconnected from the data source.
Fields	Contains a collection of Field descriptions for each column in the Recordset.

ADO.NET

- Data access component of **Microsoft's .NET** application development framework
- Two new features for development of distributed applications:
  - **DataSet** is disconnected memory-resident representation of database
  - DataSet is internally stored in XML format
    - Data in DataSet made persistent as XML documents

FIGURE 14.6 ADO.NET framework



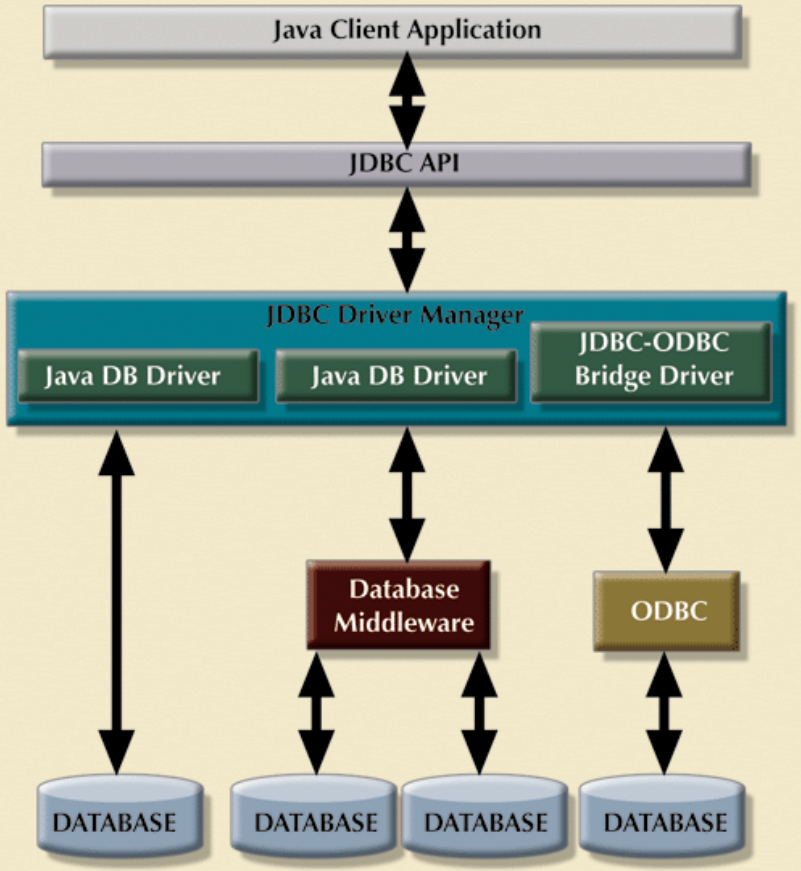
ADO.NET (cont'd.)

- Specific objects manipulate data in data source
  - Connection
  - Command
  - DataReader
  - DataAdapter
  - DataSet
  - DataTable

Java Database Connectivity (JDBC)

- **Java** is an object-oriented programming language
  - Runs on top of Web browser software
- Advantages of JDBC:
  - Company can leverage existing technology and personnel training
  - Allows direct access to database server or access via database middleware
  - Provides a way to connect to databases through an ODBC driver

FIGURE 14.7 JDBC architecture



Internet Databases

- Web database connectivity allows new innovative services that:
  - Permit rapid response by bringing new services and products to market quickly
  - Increase customer satisfaction through creation of Web-based support services
  - Yield fast and effective information dissemination through universal access

TABLE 14.3 Characteristics and Benefits of Internet Technologies

INTERNET CHARACTERISTIC	BENEFIT
Hardware and software independence	Savings in equipment/software acquisition Ability to run on most existing equipment Platform independence and portability No need for multiple platform development
Common and simple user interface	Reduced training time and cost Reduced end-user support cost No need for multiple platform development
Location independence	Global access through Internet infrastructure and mobile smart devices Reduced requirements (and costs!) for dedicated connections
Rapid development at manageable costs	Availability of multiple development tools Plug-and-play development tools (open standards) More interactive development Reduced development times Relatively inexpensive tools Free client access tools (Web browsers) Low entry costs. Frequent availability of free Web servers Reduced costs of maintaining private networks Distributed processing and scalability, using multiple servers

Web-to-Database Middleware: Server-Side Extensions

- Web server is the main hub through which Internet services are accessed
- Dynamic Web pages are at the heart of current generation Web sites
- **Server-side extension**: a program that interacts directly with the Web server
  - Also known as **Web-to-database middleware**
- Middleware must be well integrated

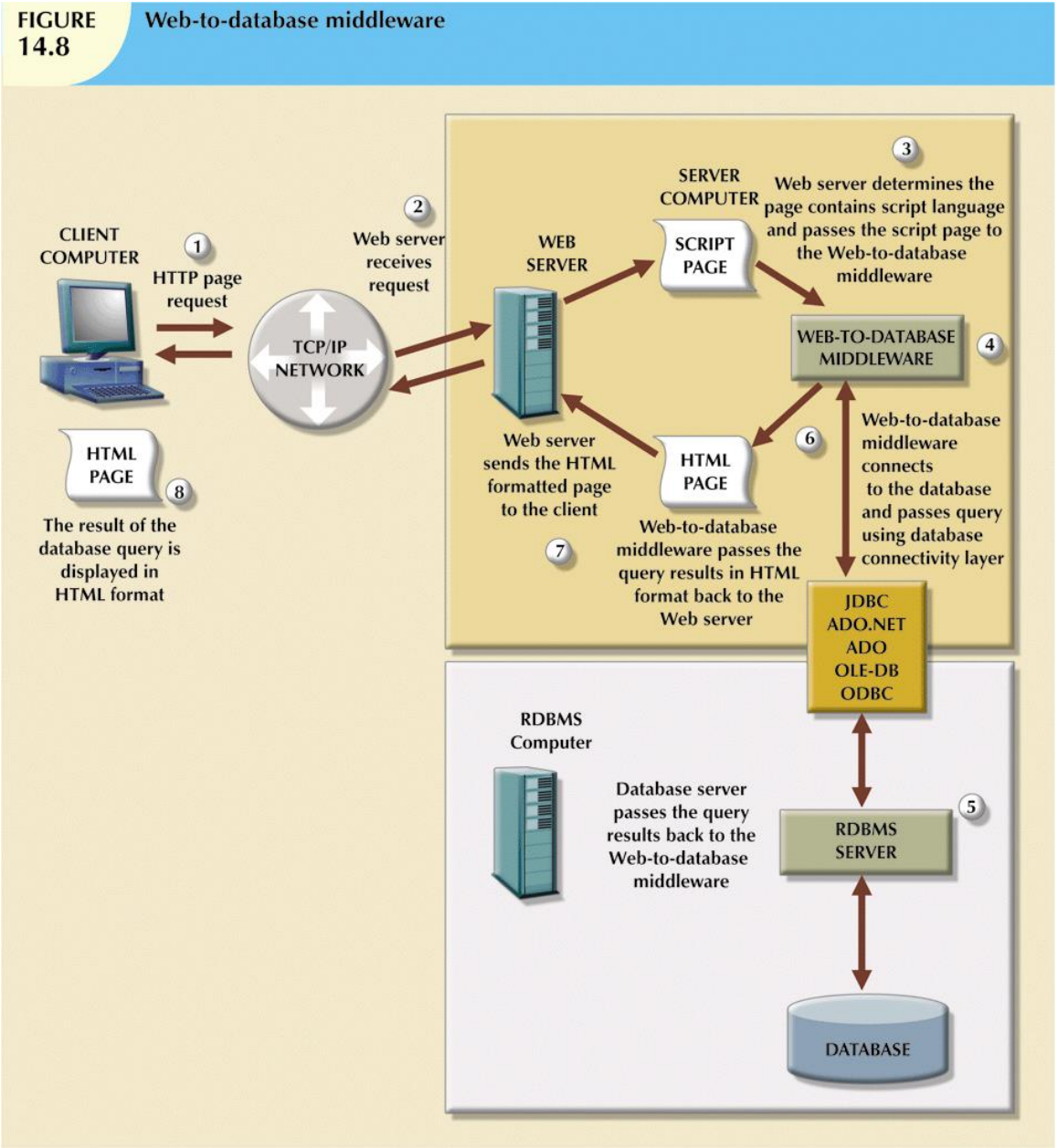
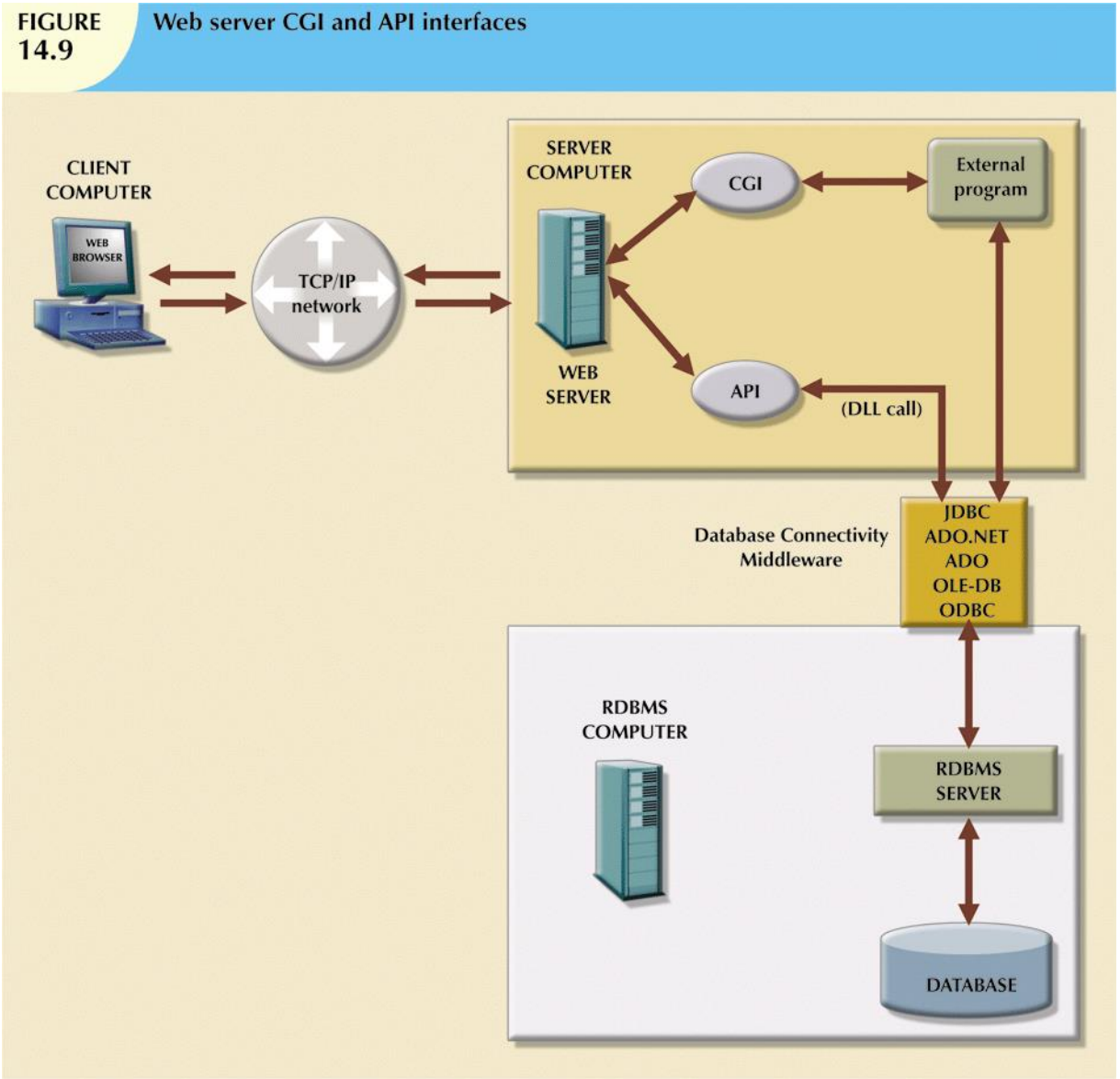


FIGURE 14.8
Web-to-database middleware
CLIENT COMPUTER
1
HTTP page request
TCP/IP NETWORK
2
Web server receives request
WEB SERVER
SERVER COMPUTER
SCRIPT PAGE
3
Web server determines the page contains script language and passes the script page to the Web-to-database middleware
WEB-TO-DATABASE MIDDLEWARE
4
Web-to-database middleware connects to the database and passes query using database connectivity layer
JDBC
ADO.NET
ADO
OLE-DB
ODBC
RDBMS Computer
RDBMS SERVER
5
DATABASE
Database server passes the query results back to the Web-to-database middleware
6
HTML PAGE
Web-to-database middleware passes the query results in HTML format back to the Web server
7
Web server sends the HTML formatted page to the client
HTML PAGE
8
The result of the database query is displayed in HTML format

Web Server Interfaces

- Two well-defined Web server interfaces:
  - **Common Gateway Interface (CGI)**
  - **Application Programming Interface (API)**
- Disadvantage of CGI scripts:
  - Loading external script decreases system performance
  - Language and method used to create script also decrease performance
- API is more efficient than CGI
  - API is treated as part of Web server program

FIGURE 14.9 Web server CGI and API interfaces



The Web Browser

- Software that lets users navigate the Web
- Located in client computer
- Interprets HTML code received from Web server
- Presents different page components in standard way
- Web is a **stateless system**: Web server does not know the status of any clients

Client-Side Extensions

- Add functionality to Web browser
- Three general types:
  - Plug-ins
  - Java and JavaScript
  - ActiveX and VBScript

Client-Side Extensions (cont'd.)

- **Plug-in**: an external application automatically invoked by the browser when needed
- Java and **JavaScript**: embedded in Web page
  - Downloaded with the Web page and activated by an event
- **ActiveX** and **VBScript**: embedded in Web page
  - Downloaded with page and activated by event
  - Oriented to Windows applications

Web Application Servers

- Middleware application that expands the functionality of Web servers
  - Links them to a wide range of services
- Some uses of Web application servers:
  - Connect to and query database from Web page
  - Create dynamic Web search pages
  - Enforce referential integrity
- Some features of Web application servers:
  - Security and user authentication
  - Access to multiple services

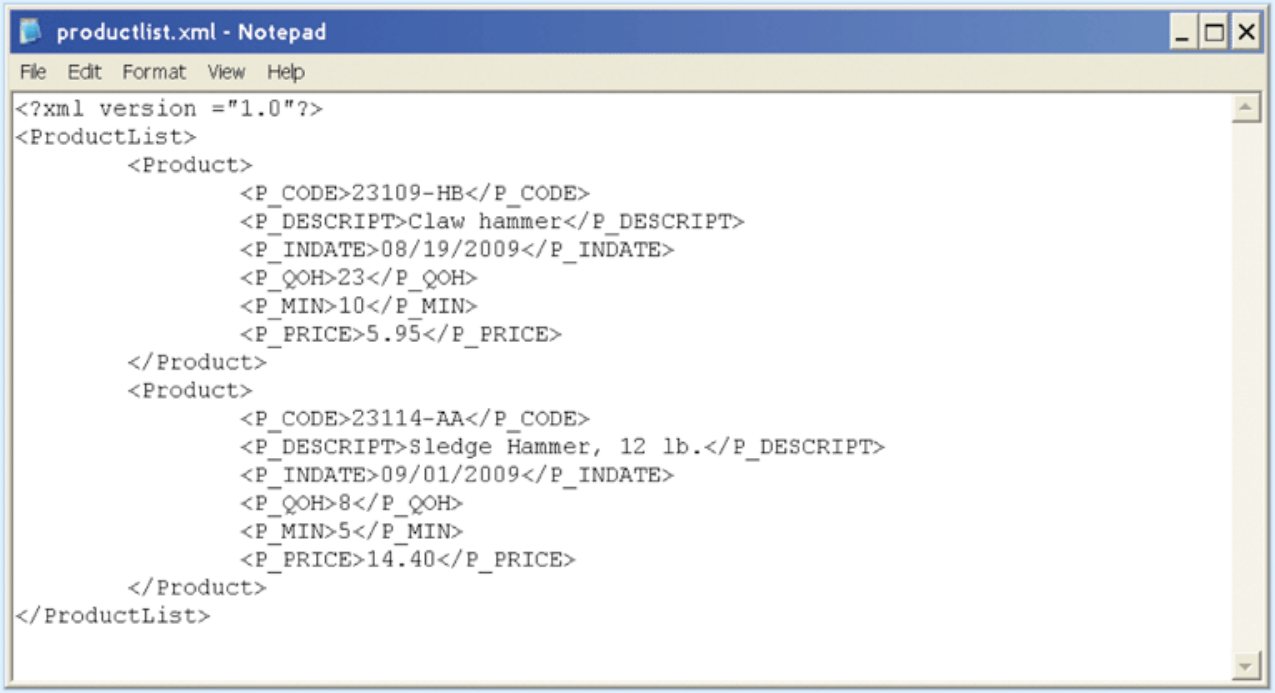
Extensible Markup Language (XML)

- Companies use Internet to create new systems that integrate their data
  - Increase efficiency and reduce costs
- Electronic commerce enables organizations to market to millions of users
- Most e-commerce transactions take place between businesses
- HTML Web pages display in the browser
  - Tags describe how something looks on the page

Extensible Markup Language (XML) (cont'd.)

- **Extensible Markup Language (XML)**:
  - Metalanguage to represent and manipulate data elements
  - Facilitates exchange of structured documents over the Web
  - Allows definition of new tags
    - Case sensitive
    - Must be well-formed and properly nested
    - Comments indicated with <- and ->
    - *XML* and *xml* prefixes reserved for XML tags only

FIGURE 14.10 Contents of the productlist.xml document



Document Type Definitions (DTD) and XML Schemas

- **Document Type Definition (DTD)**
  - File with .dtd extension that describes elements
  - Provides composition of database's logical model
  - Defines the syntax rules or valid tags for each type of XML document
- Companies engaging in e-commerce transaction must develop and share DTDs
- DTD referenced from inside XML document

FIGURE 14.11 Contents of the productlist.dtd document

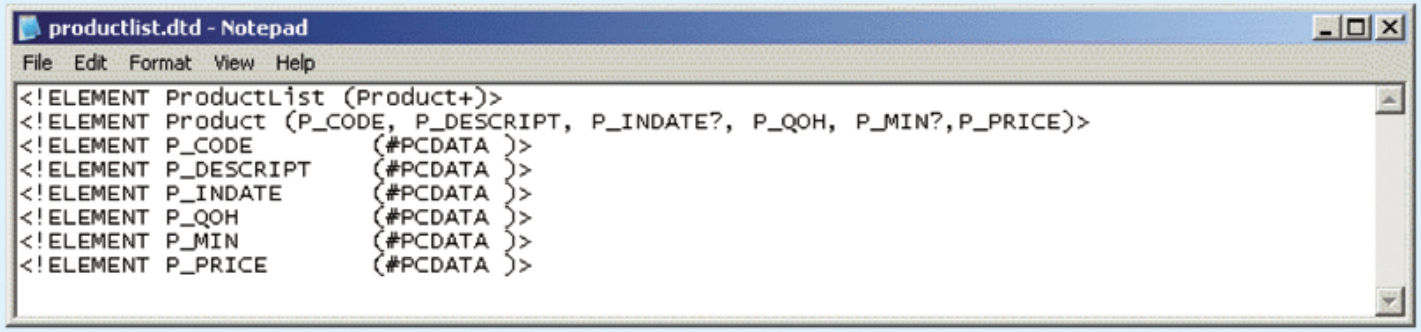


FIGURE 14.12 Contents of the productlistv2.xml document

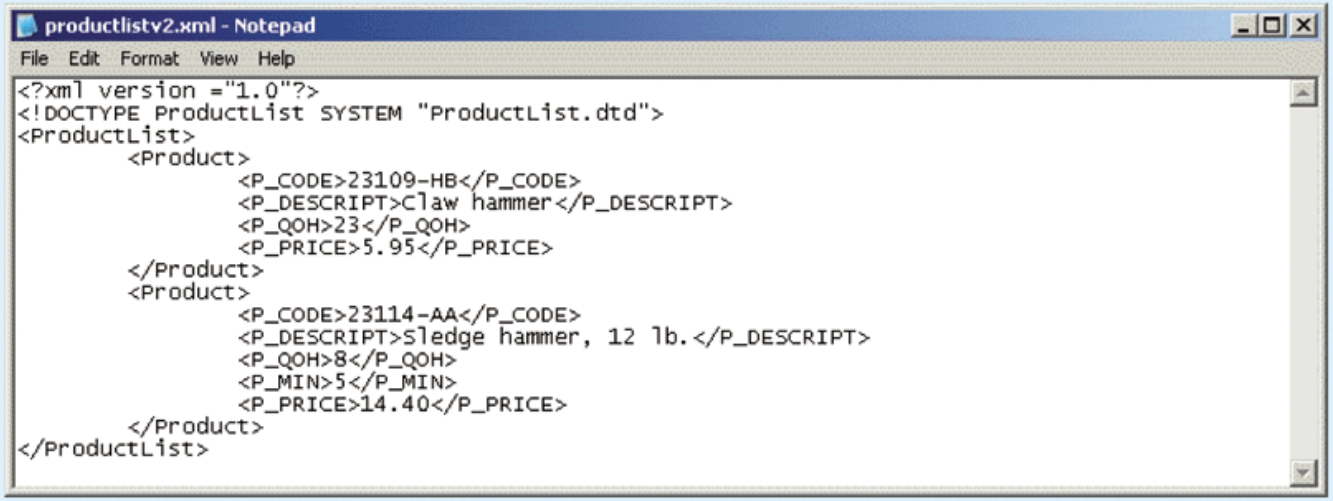
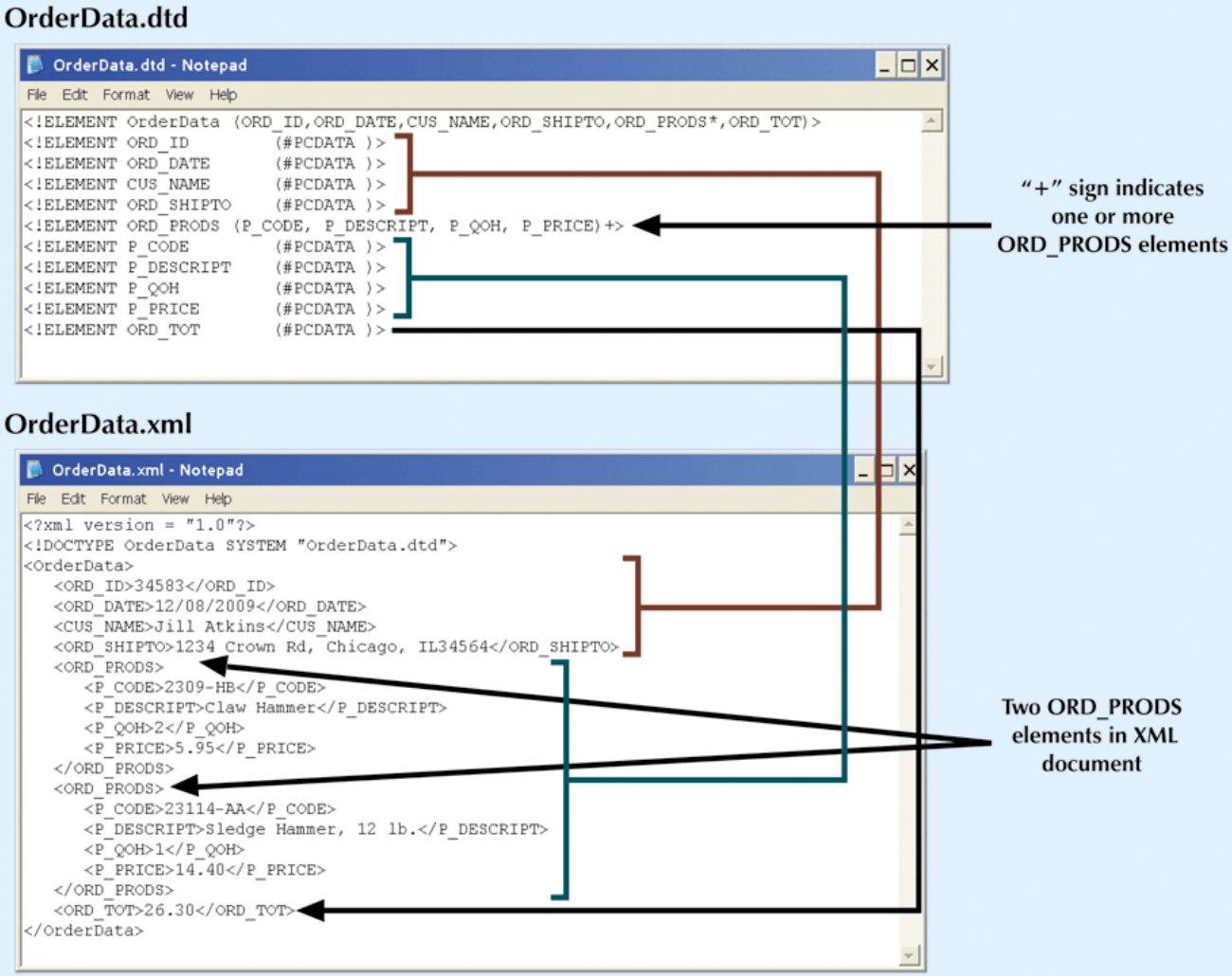


FIGURE 14.13 DTD and XML documents for order data



Document Type Definitions (DTD) and XML Schemas (cont'd.)

- **XML schema**
  - Advanced data definition language
  - Describes the structure of XML data documents
- Advantage of XML schema:
  - More closely maps to database terminology and features
- **XML schema definition (XSD)** file uses syntax similar to XML document

FIGURE 14.14 The XML schema document for the order data

OrderData.xsd - Notepad

File Edit Format View Help

```
<xsd:schema xmlns:xsd="http://www.company.com/xmlschema">

<xsd:element name="OrderData" type="Order"/>

<xsd:complexType name="Order">
        xsd:element name="ORD_ID"       type="xsd:string/>
        xsd:element name="ORD_DATE"     type="xsd:date/>
        xsd:element name="ORD_NAME"     type="xsd:string/>
        xsd:element name="CUS_NAME"     type="xsd:string/>
        xsd:element name="ORD_SHIPTO"   type="xsd:string/>
        xsd:element name="ORD_PRODS"    type="xsd:productlist/>
        xsd:element name="ORD_TOT"      type="xsd:decimal/>
</xsd:complexType name="Order">

<xsd:complexType name="prodlist"
        <xsd:element name="product"     type="aproduct" minOccurs="1" maxOccurs="unbounded"/>
</xsd:complexType>

<xsd:complexType name="aproduct">
        xsd:element name="P_CODE"       type="xsd:string"               use="required"/>
        xsd:element name="P_DESCRIPT"   type="xsd:string"               use="required"/>
        xsd:element name="P_QOH"        type="xsd:positiveInteger"      use="required"/>
        xsd:element name="P_PRICE"      type="xsd:decimal               use="required"/>
</xsd:complexType>

</xsd:schema>
```

XML Presentation

- XML separates data structure from presentation and processing
- Extensible Style Language (XSL) displays XML data
  - Defines the rules by which XML data are formatted and displayed
  - Two parts:
    - Extensible Style Language Transformations (XSLT)
    - XSL style sheets

FIGURE 14.15 Framework for XML transformations

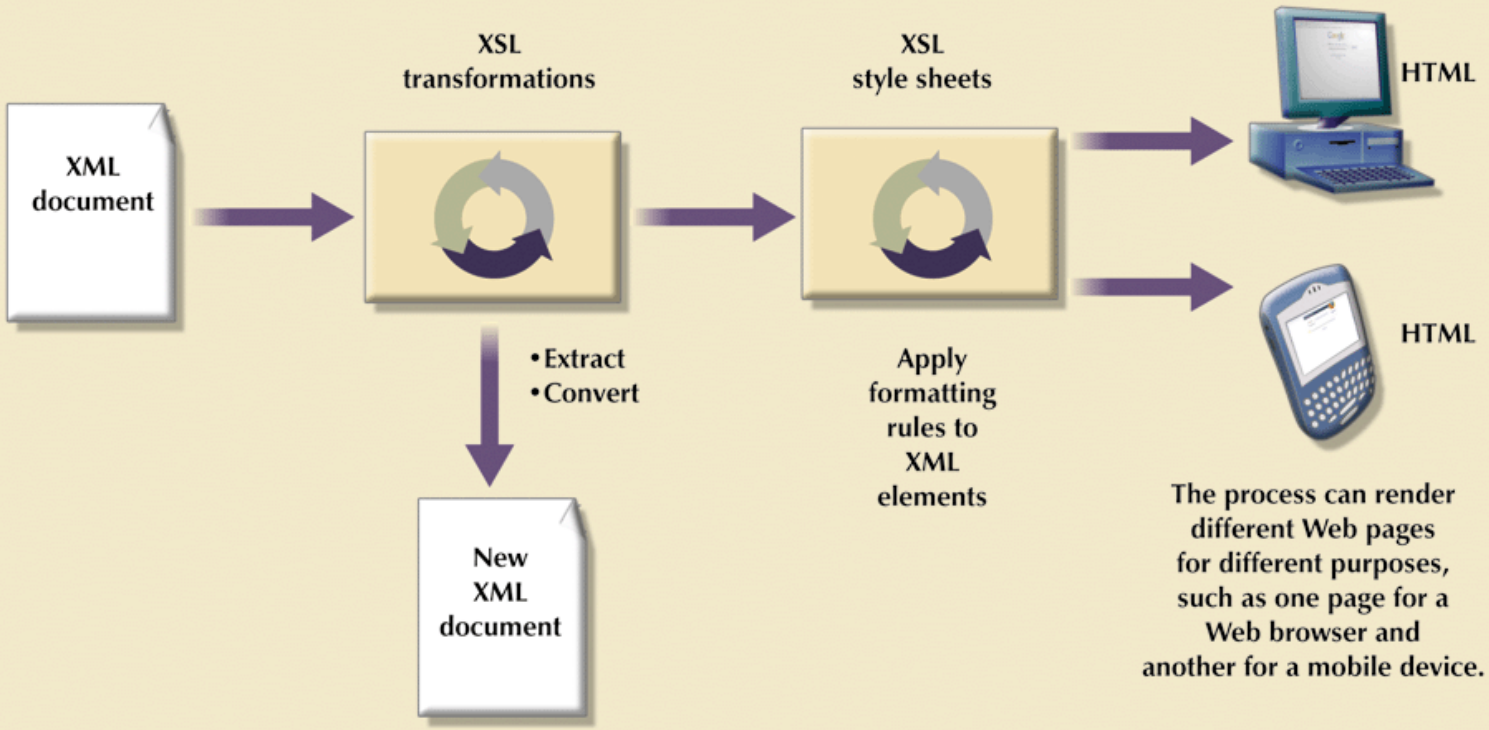


FIGURE 14.16 Displaying XML documents

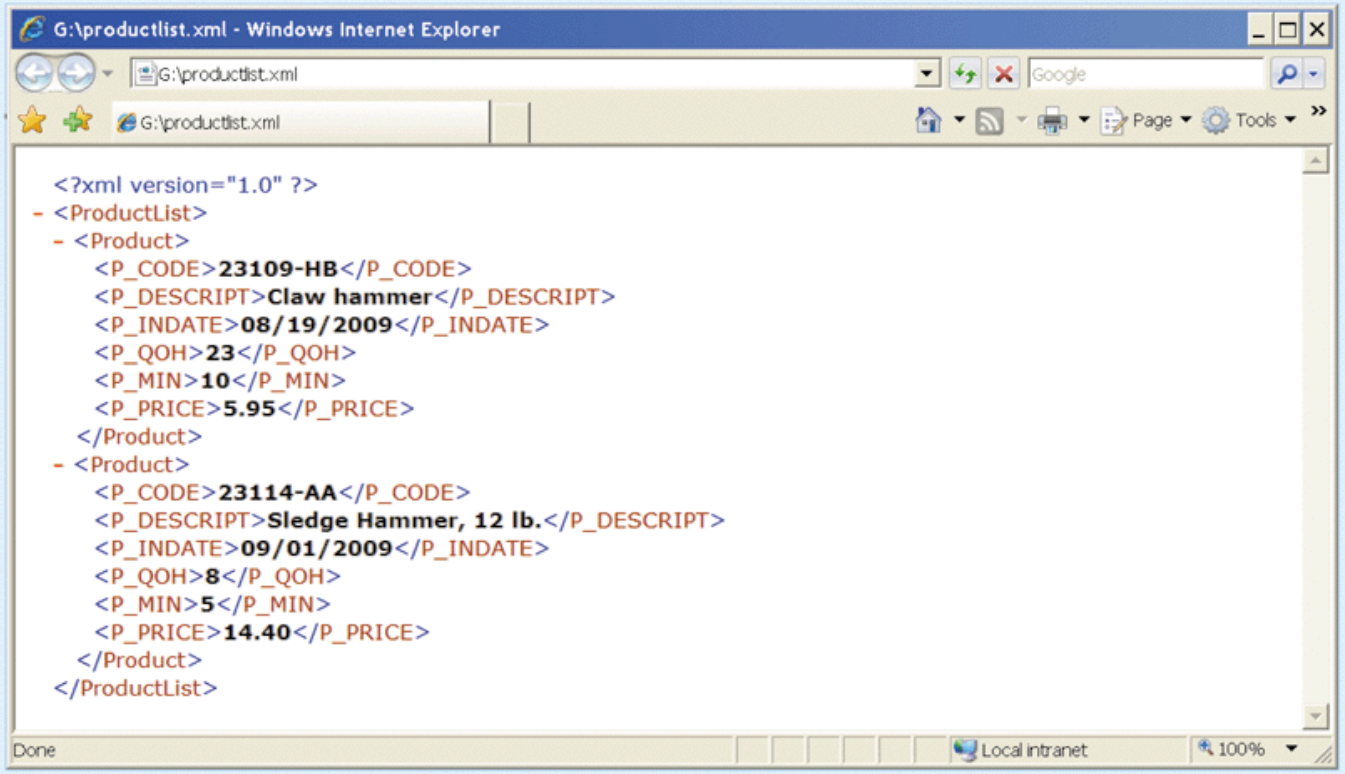
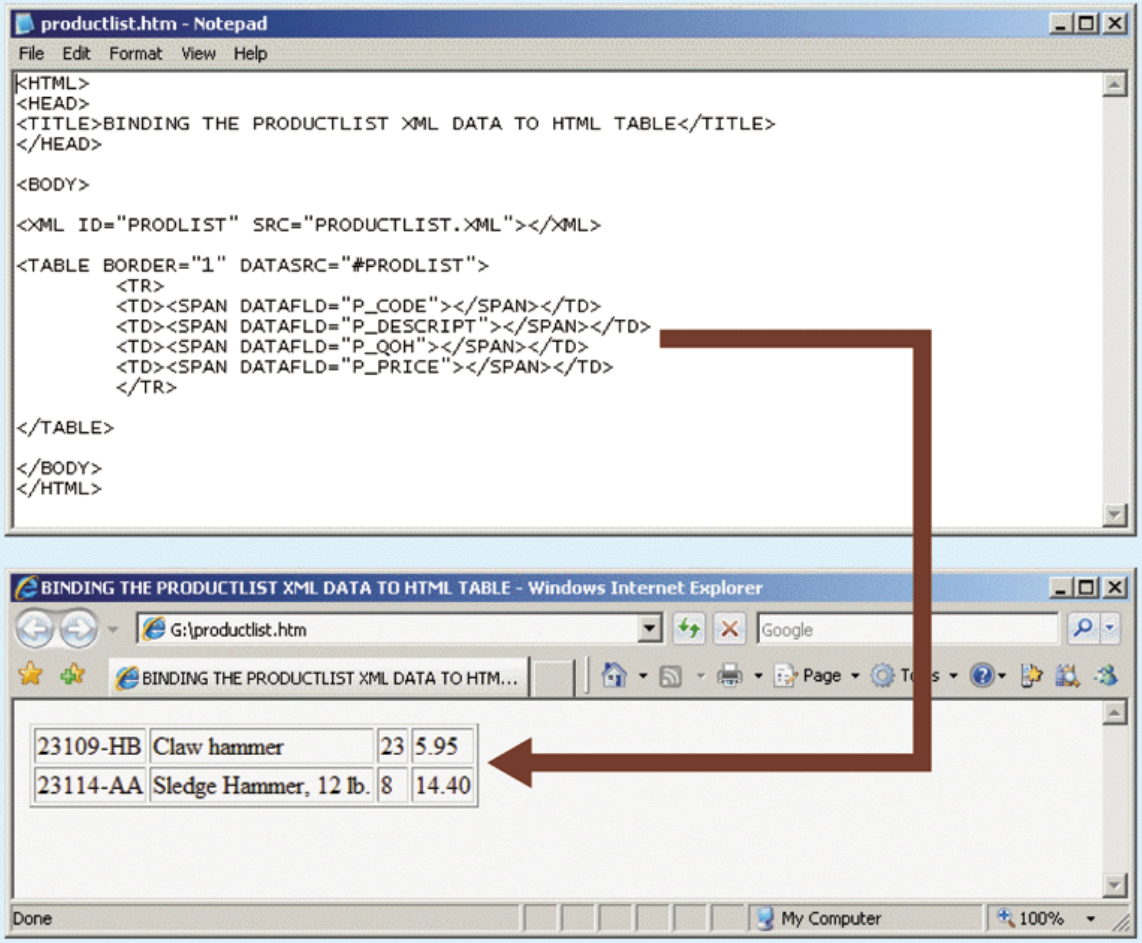


FIGURE 14.17 XML data binding



XML Applications

- B2B exchanges
- Legacy systems integration
- Web page development
- Database support
- Database meta-dictionaries
- XML databases
- XML services

SQL Data Services

- Provides relational data management to companies of any size
- Avoids high cost of personnel/maintenance
- Leverages Internet to provide:
  - Hosted data management
  - Standard protocols
  - A common programming interface
- Could assist businesses with limited information technology resources