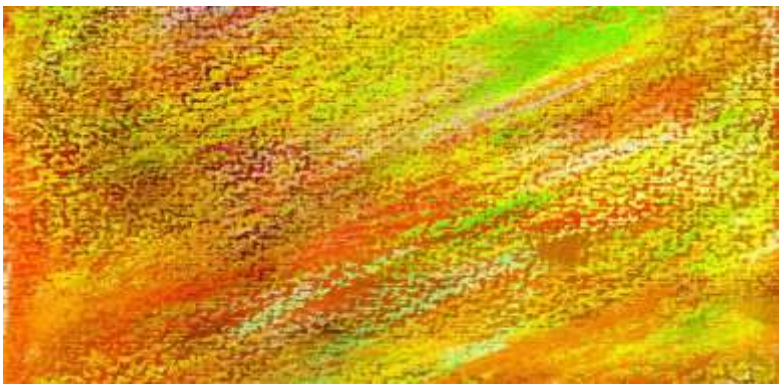
DAVID M. KROENKE and DAVID J. AUER

DATABASE CONCEPTS, 7th Edition

Chapter Seven

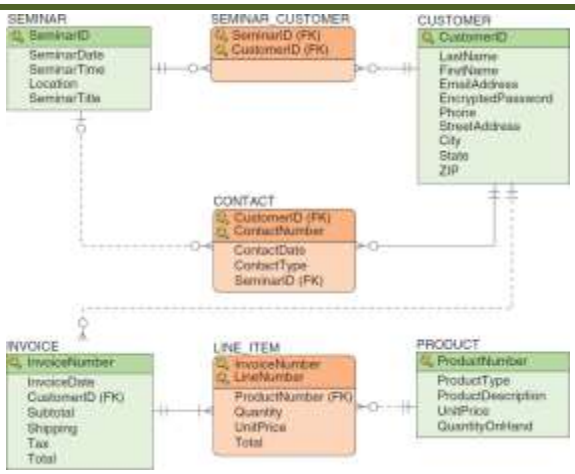
Database Processing Applications



Chapter Objectives

- Understand and be able to set up Web database processing
- Learn the basic concepts of Extensible Markup Language (XML)

Heather Sweeney Designs:
Database Design



Heather Sweeney Designs:
HSD Database Diagram in SQL Server 2014

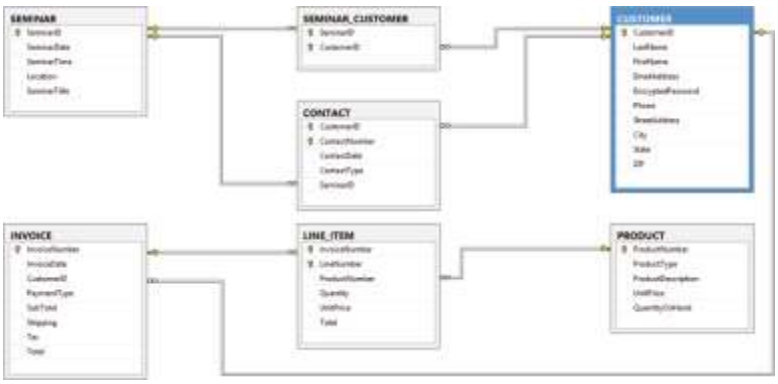


Figure 7-1: The HSD Database Diagram

The Database Processing Environment I

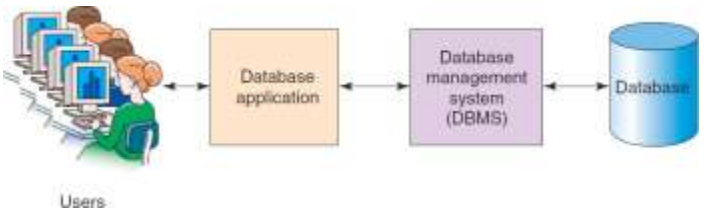


Figure 7-2: Components of a Database System

The Database Processing Environment II

- A database processing environment is complicated and multi-faceted.
- It has:
  - Multiple users
  - Multiple queries
  - Multiple forms
  - Multiple reports
  - Multiple application programs

The Database Processing Environment

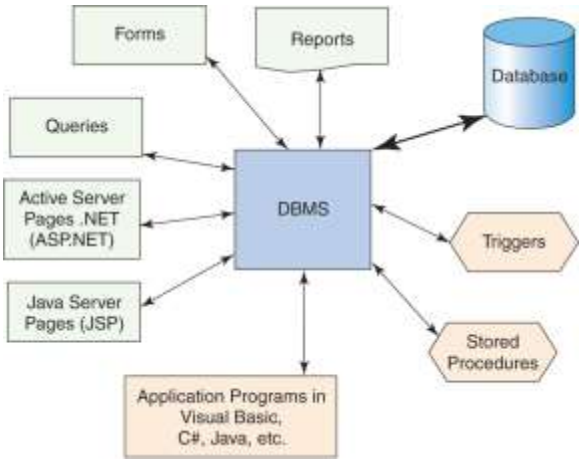


Figure 7-3: The Database Processing Environment

Processing Constraints

- Enforcing referential integrity
- Cascading deletion
- Cascading modifications
- Data type constraints
- Data size constraints
- Data value constraints
- Null constraints
- Uniqueness constraints

Triggers and Stored Procedures

- A **trigger** is a stored procedure that is automatically invoked by the DBMS when a specified activity occurs.
  - BEFORE, AFTER, and INSTEAD OF
- A **stored procedure** is a module similar to the subroutine or function that performs database actions.
  - Stored in the database itself

Internet Application Processing

- Internet Application Processing is more complicated than traditional application processing.
- Specifically, with Internet Application Processing
  - The network becomes an integral part of the application.

Internet Application Processing Environment

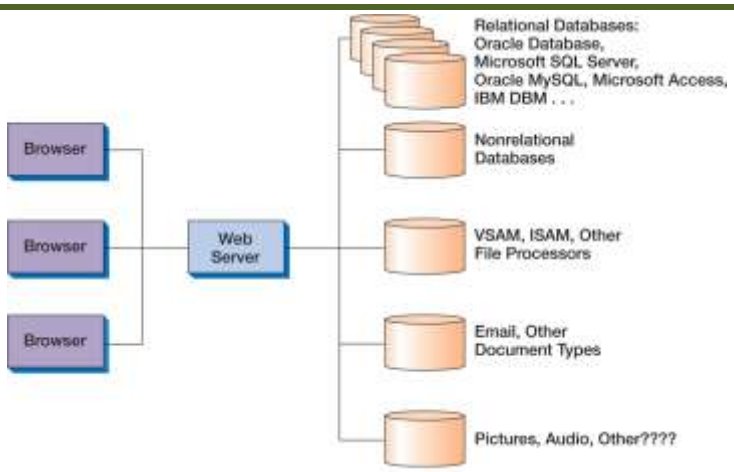


Figure 7-4: The Web Database Processing Environment

APIs

- Every DBMS product has an **application programming interface (API).**
  - An API is a collection of objects, methods and properties for executing DBMS functions from program code.
  - Each DBMS has its own API, and APIs vary from one DBMS product to another.
- To simplify this situation, the computer industry has developed standards for database access.
- API interface standards are used to make it easier for programmers—they write to the interface standard instead of the DBMS API.
  - ODBC – Open Database Connectivity
  - JDBC – Java Database Connectivity

OBDC and OLE DB

- The **Open Database Connectivity (ODBC)** standard was developed in the early 1990s.
  - Provides a DBMS-independent means for processing relational database data
- **OLE DB** was created by Microsoft in the mid-1990s.
  - An object-oriented interface
  - Encapsulates data-server functionality
  - Designed not just for access to relational databases, but also for accessing many other types of data as well
  - Readily accessible to programming languages, such as C, C#, and Java
  - Not as accessible to Visual Basic and scripting languages

ADO and ADO.NET

- Microsoft developed **Active Data Objects (ADO)**.
  - A set of objects for utilizing OLE DB
  - Designed for use by any language, including VB, VBScript, and Jscript
- ADO has been followed by **ADO.NET**.
  - An improved version of ADO was developed as part of Microsoft's .NET initiative.

The Role of ADO.NET
in the Web Database Processing Environment

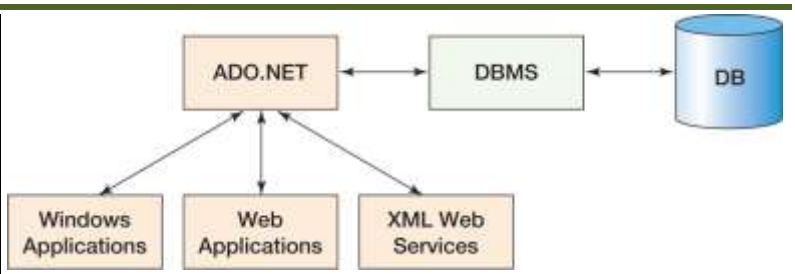


Figure 7-5: The Role of ADO.NET

N-Tier Architecture

- **Tiers** refer to the number of computers involved in the Web database application.
  - The workstation with Web browser is the first tier.
  - **Two-tier architecture** means that the Web server and the DBMS are on the same server.
  - **Three-tier architecture** means that the Web server and the DBMS are on separate servers.

AMP, LAMP and WAMP

- **AMP** refers to a combination of the Apache Web server, the MySQL DBMS, and the PHP or Pearl programming language.
- **LAMP** is AMP running on the Linux operating system.
- **WAMP** is AMP running on the Windows operating system.

Typical Web Server Setups

	Microsoft Windows	Linux or Unix
Web Server	Internet Information Server (IIS)	Apache
Scripting Languages (Interpreted)	ASPX Pages with VBScript or JScript (PHP also works)	PHP
Object Oriented Languages (Complied)	ASP.NET pages with VisualBasic.Net, C#.Net or others	JSP pages with Java (Also see the Mono project)

Typical Web DBMS Servers

	Microsoft Windows	Linux or Unix
Web Server	Internet Information Server (IIS)	Apache
DBMS	Microsoft SQL Server	Oracle Database Oracle MySQL
API Interface Standard	ODBC	JDBC

ODBC: Three-tier Web Server Architecture

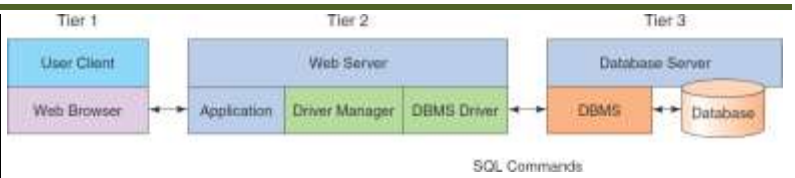


Figure 7-6: ODBC Three-Tier Web Server Architecture

ODBC Architecture

- The *application program*, the *ODBC driver manager*, and the *ODBC DBMS driver* all reside on the Web server.
- The DBMS driver sends requests to *data sources* on the database server.
  - A **data source** is the database, its associated DBMS, operating system, and network platform.

ODBC Architecture: The Application Program

- The **application program** issues requests:
  - To create a connection with a data source
  - To issue SQL statements and receive results
  - To process errors
  - To start, commit and roll back transactions
- ODBC
  - Provides a standard means for each of these requests
  - Defines a standard set of error codes and messages

ODBC Architecture: The ODBC Driver Manager

- The **ODBC driver manager**
  - Serves as an intermediary between the application and the DBMS drivers
  - When the application requests a connection, the driver:
    - Determines the type of DBMS that processes a given ODBC data source
    - Loads the appropriate driver into memory (if it is not already loaded)

ODBC Architecture: The ODBC Driver

- An **ODBC driver**
  - Processes ODBC requests
  - Submits specific SQL statements to a given type of data source
    - There is a different driver for each data source type.
    - It is the responsibility of the driver to ensure that standard ODBC commands execute correctly.
  - Converts data source error codes and messages into the ODBC standard codes and messages

ODBC Architecture: ODBC Data Sources

- An **ODBC data source i**s an ODBC data structure that identifies a database and the DBMS that processes it.
  - There are three types of data sources.
    - A **file data source** is a file that can be shared among database users—the only requirement is that the users have the same DBMS driver and privilege to access the database.
    - A **system data source** is one that is local to a single computer—the operating system and any user on that system (with proper permissions) can use a system data source.
    - A **user data source** is available only to the user who created it.
  - Each created data source is given a **data source name (DSN)** that is used to reference the data source.

ODBC Architecture: Creating an ODBC Data Source I

Select System DSN and click the **Add** button

Select **ODBC Driver 11 for SQL Server**

Click the **Finish** button

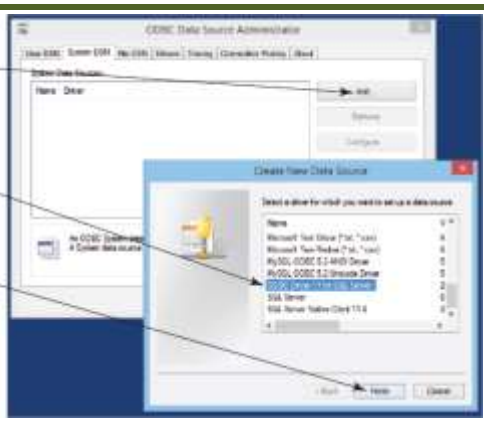
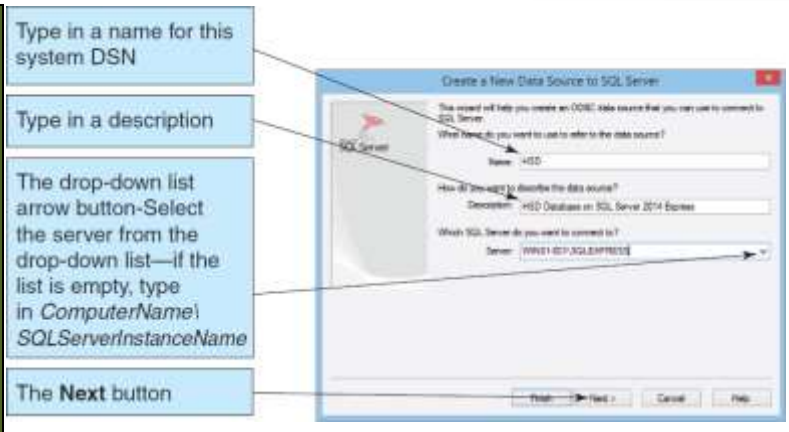


Figure 7-7: The Create New Data Source Dialog Box

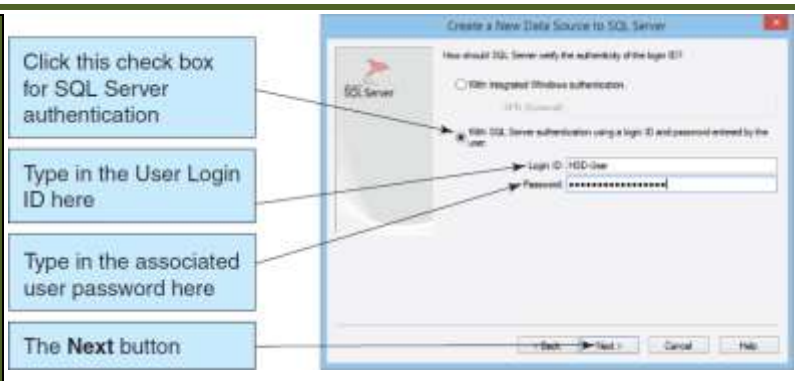
ODBC Architecture: Creating an ODBC Data Source II



(a) Naming the ODBC Data Source

Figure 7-8: The Create New Data Source to SQL Server Dialog Box

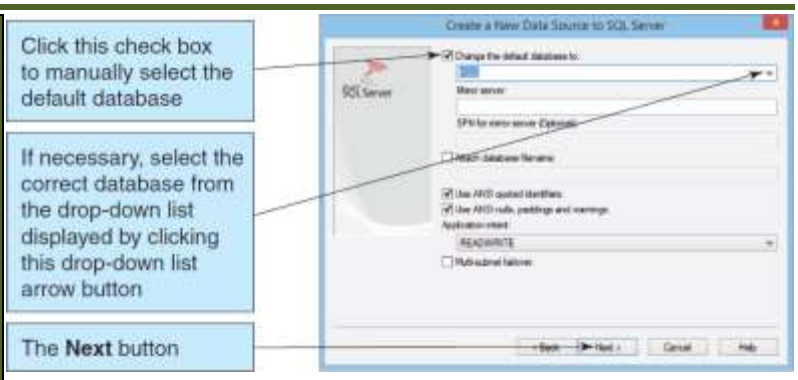
ODBC Architecture: Creating an ODBC Data Source III



(b) Selecting the User Login ID Authentication Method

Figure 7-8:
The Create New Data Source to SQL Server Dialog Box (Cont'd)

ODBC Architecture: Creating an ODBC Data Source IV



(c) Selecting the Default Database

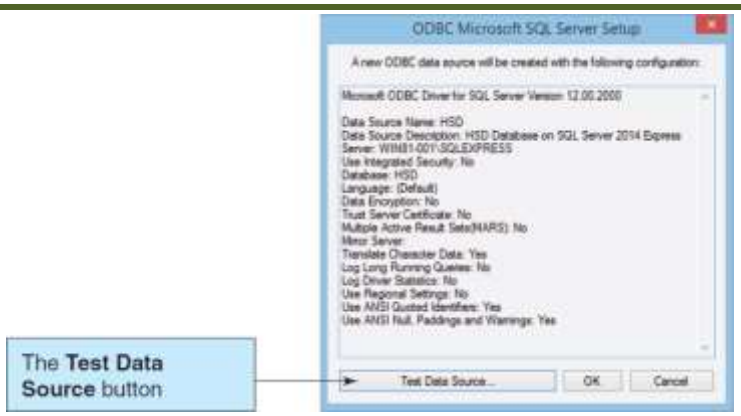
Figure 7-8:
The Create New Data Source to SQL Server Dialog Box (Cont'd)

ODBC Architecture: Creating an ODBC Data Source V

(d) Additional Setting Options

Figure 7-8:
The Create New Data Source to SQL Server Dialog Box (Cont'd)

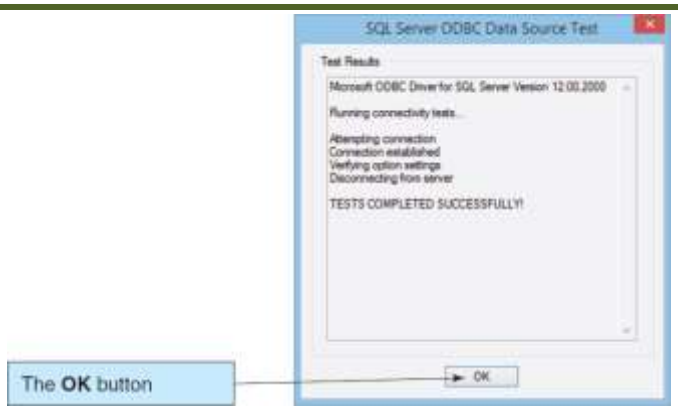
ODBC Architecture: Creating an ODBC Data Source VI



(e) Testing the Data Source

Figure 7-8:
The Create New Data Source to SQL Server Dialog Box (Cont'd)

ODBC Architecture: Creating an ODBC Data Source VII



(f) The Successfully Tested Data Source

Figure 7-8:
The Create New Data Source to SQL Server Dialog Box (Cont'd)

ODBC Architecture: The Completed ODBC Data Source

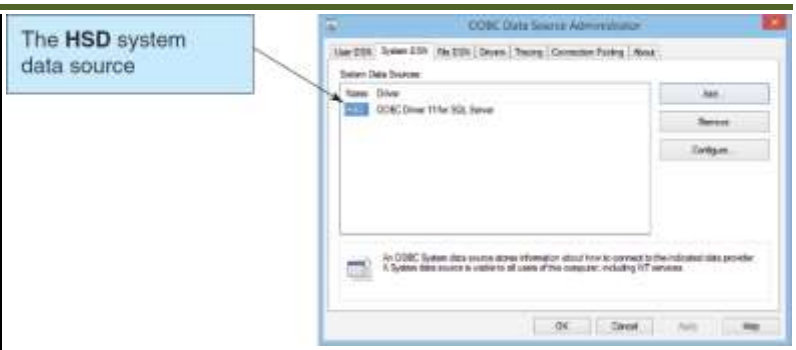


Figure 7-9: The Completed HSD System Data Source

32-Bit versus 64-Bit ODBC I

- The ODBC Data Sources program we have been using is for the *64-bit* version of the ODBC data sources administration program, and will create *64-bit ODBC data sources*.
- These will ***only*** work as long as ***every*** component in the Web application program chain is a 64-bit program. But, if ***any*** component is a 32-bit program, the 64-bit ODBC data source will ***not*** work.

32-Bit versus 64-Bit ODBC II

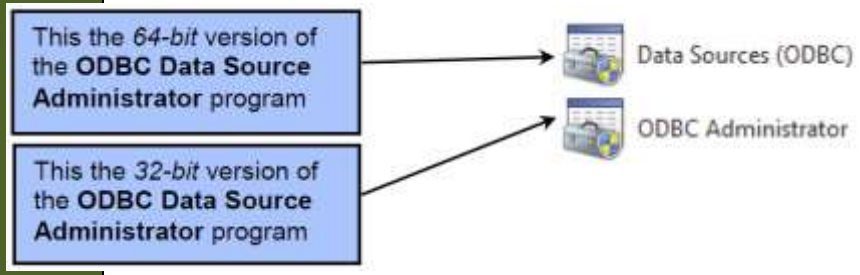
- In fact, when you are building the Web database applications in this chapter, if everything seems to be correctly done but the system still does not work, the most likely cause is a 32-bit program in the chain.
- To resolve this problem, you will need to use the *32-bit* version of the ODBC Data Sources program located at *C:\Windows\SysWOW64\odbcad32.exe*.

32-Bit versus 64-Bit ODBC III

- To resolve your Web database application problem, create a 32-bit ODBC data source using OBDC Administrator, and then revise your Web page code to use that ODBC data source.
- For more information of the programs in the Windows\SysWOW64 folder, see the Wikipedia article on WoW64 at http://en.wikipedia.org/wiki/WoW64.

32-Bit versus 64-Bit ODBC IV

- The two icons for the two versions appear as shown below. Note that although the program icons are the same, the program names are different. Data Source (OBDC) is the 64-bit version, and OBDC Administrator is the 32-bit



IIS: The *wwwroot* Folder

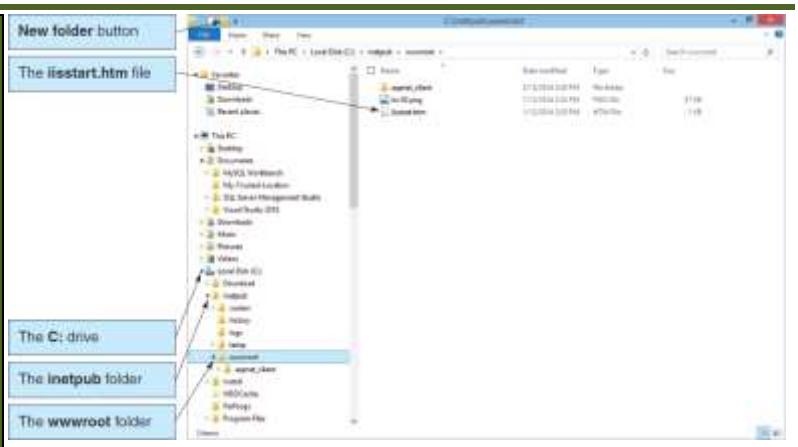


Figure 7-10: The IIS wwwroot Folder

IIS:
The IIS Management Program

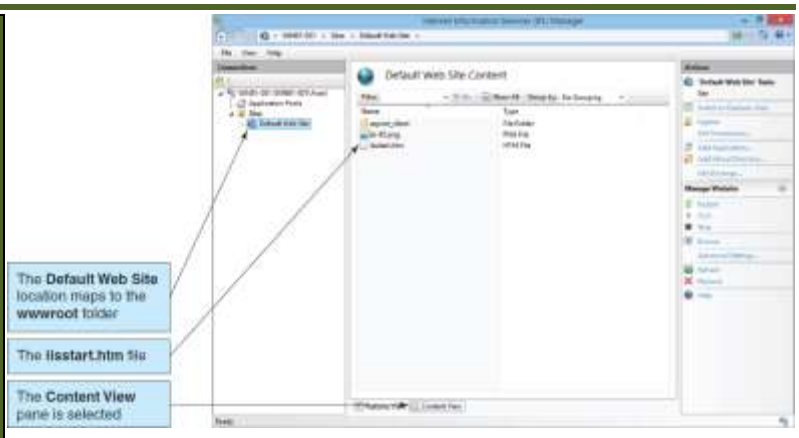


Figure 7-11: Managing IIS

IIS:
The IIS Localstart Web Page

Figure 7-12: The Default IIS Web Page

IIS:
The index.html Default Document

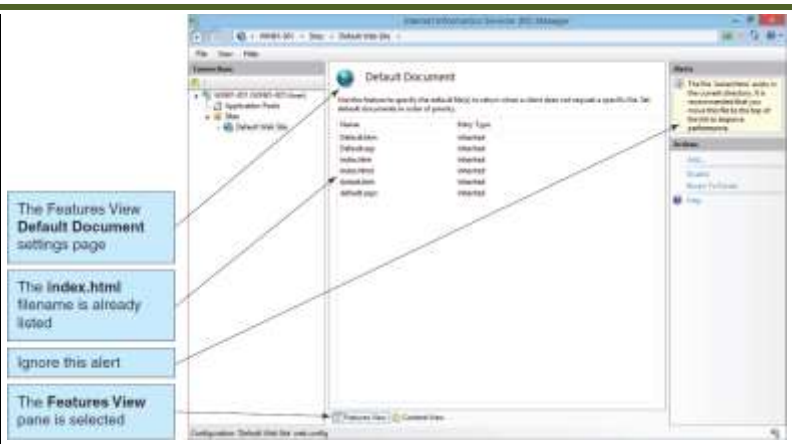


Figure 7-13: The index.html File in Windows 7 IIS Manager

Web Pages:
HTML Code for *index.html*

```
<!DOCTYPE html>
<html>
    <head>
        <title>Heather Sweeney Designs Demonstration Pages Home Page</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <h1 style="text-align: center; color: blue">
            Database Concepts (7th Edition)
        </h1>
        <p style="text-align: center; font-weight: bold">
            David M. Kroenke
        </p>
        <p style="text-align: center; font-weight: bold">
            David J. Auer
        </p>
        <hr />
        <h1 style="text-align: center; color: blue">
            Welcome to the Heather Sweeney Designs Home Page
        </h1>
        <hr />
        <p>Chapter 7 Demonstration Pages From Figures in the Text:</p>
        <p>Example 1:   
            <a href="ReadSeminar.php">
                Display the SEMINAR Table (No surrogate key)
            </a>
        </p>
        <hr />
    </body>
</html>
```

HTML Code for *index.html* in Notepad

The **index.html** HTML code—note how indentation is used to keep the code organized and readable

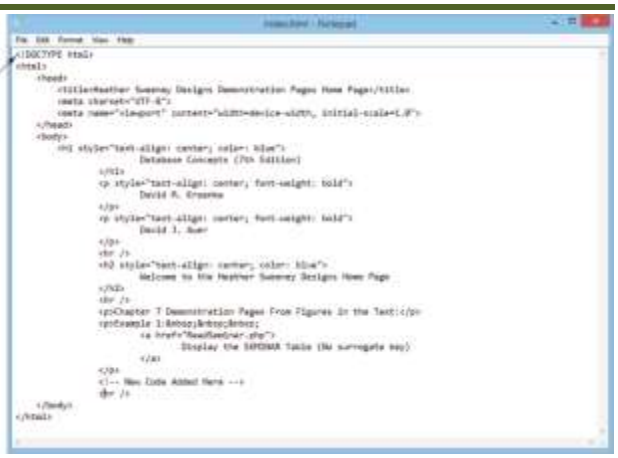


Figure 7-15: The HTML Code for the index.html File in Notepad

Web Pages: The *default.htm* Web Page

Figure 7-16: The index.html Web Page in HSD

HTML Code for *index.html* in the NetBeans IDE

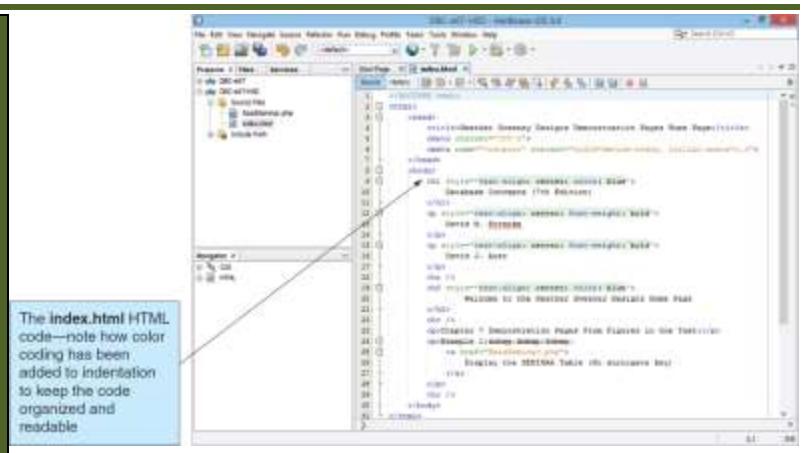


Figure 7-17:
The HTML Code for the index.html File in the NetBeans IDE

The *ReadSeminar.php* Code: Page Setup and Style Section

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>ReadSeminar PHP Page</title>
        <style type="text/css">
            h1 {text-align: center; color: blue}
            h2 {font-family: Ariel, sans-serif; text-align: left; color: blue}
            p.footer {text-align: center}
            table.output {font-family: Ariel, sans-serif}
        </style>
    </head>
```

From Figure 7-18: The HTML and PHP Code for ReadSeminar.php

The *ReadSeminar.php* Code: Creating A Connection

```
<body>
<?php
        // Get connection
        $Conn = odbc_connect('HSD', 'HSD-User','HSD-User+password');

        // Test connection
        if (!Conn)
                {
                exit ("ODBC Connection Failed: " . $Conn);
                }
```

From Figure 7-18: The HTML and PHP Code for ReadSeminar.php

The *ReadSeminar.php* Code: Creating A Recordset

```
        // Create SQL statement
        $SQL = "SELECT * FROM SEMINAR";

        // Execute SQL statement
        $RecordSet = odbc_exec($Conn,$SQL);

        // Test existence of recordset
        if (!$RecordSet)
                {
                exit ("SQL Statement Error! " . $SQL);
                }
?>
```

From Figure 7-18: The HTML and PHP Code for ReadSeminar.php

The *ReadSeminar.php* Code: Displaying the Results

```
<!-- Page Headers -->
<h1>
    The Heather Sweeney Designs SEMINAR Table
</h1>
<hr />
<h2>
    SEMINAR
</h2>
<?php

    // Table headers
    echo "<table class='output' border='1'>
        <tr>
            <th>SeminarDate</th>
            <th>SeminarTime</th>
            <th>Location</th>
            <th>SeminarTitle</th>
        </tr>";

    // Table data
    while($RecordSetRow = odbc_fetch_array($RecordSet))
        {
        echo "<tr>";
        echo "<td>" . $RecordSetRow['SeminarDate'] . "</td>";
        echo "<td>" . $RecordSetRow['SeminarTime'] . "</td>";
        echo "<td>" . $RecordSetRow['Location'] . "</td>";
        echo "<td>" . $RecordSetRow['SeminarTitle'] . "</td>";
        echo "</tr>";
        }
    echo "</table>";
```

From Figure 7-18: The HTML and PHP Code for ReadSeminar.php

The *ReadSeminar.php* Code: Disconnecting from the Database

```
    // Close connection
    odbc_close($Conn);
?>
    <hr />
    <hr />
    <p class="footer">
        <a href="../HSD/index.html">
            Return to Heather Sweeney Designs Home Page
        </a>
    </p>
    <hr />
</body>
</html>
```

From Figure 7-18: The HTML and PHP Code for ReadSeminar.php

The *ReadSeminar.php* PHP Code In the NetBeans IDE

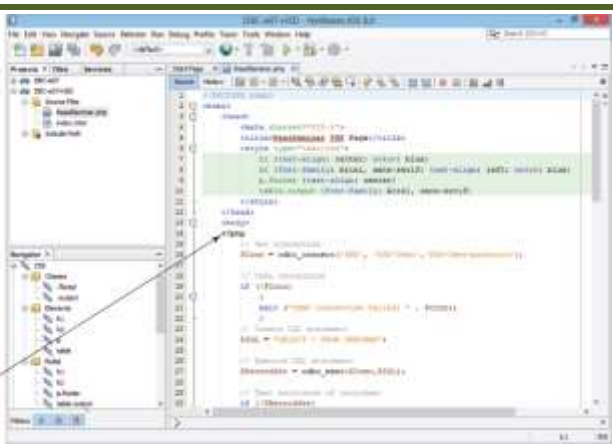


Figure 7-19:
The HTML and PHP Code for ReadSeminar.php in the NetBeans IDE

The *ReadSeminar.php* Web Page

The Heather Sweeney Designs SEMINAR Table

SEMINAR

SeminarDate	SeminarTime	Location	SeminarTitle
2013-10-12	11:00:00.0000000	San Antonio Convention Center	Kitchen on a Budget
2013-10-26	16:00:00.0000000	Dallas Convention Center	Kitchen on a Big D Budget
2013-11-02	08:30:00.0000000	Austin Convention Center	Kitchen on a Budget
2014-03-22	11:00:00.0000000	Dallas Convention Center	Kitchen on a Big D Budget
2014-03-23	11:00:00.0000000	Dallas Convention Center	Kitchen on a Big D Budget
2014-04-05	08:30:00.0000000	Austin Convention Center	Kitchen on a Budget

Return to Heather Sweeney Designs Home Page

Figure 7-20: The Results of ReadSeminar.php

The Modified Home Page

Figure 7-22: The Modified HSD Web Home Page

The NewSeminarForm Web Page

Figure 7-23: The NewSeminarForm Web Page

Entering Data in the NewSeminarForm Web Page

Figure 7-24: Entering Data Values in the NewSeminarForm Web Page

New Data in the SEMINAR Table

Figure 7-25: The New Seminar Data in the SEMINAR Table

Creating Drop-down Boxes

```
<p>
    <b>Select Seminar Location:</b>
</p>
<select name="Location">
    <option value="Austin Convention Center">Austin Convention Center</option>
    <option value="Dallas Convention Center">Dallas Convention Center</option>
    <option value="Fort Worth Convention Center">Fort Worth Convention Center</option>
    <option value="Houston Convention Center">Houston Convention Center</option>
    <option value="San Antonio Convention Center">San Antonio Convention Center</option>
</select>
<br />
```

From Figure 7-26: The HTML Code for NewSeminarForm.html

PHP POST METHOD

On *NewSeminarForm.html* page

```
<form action="InsertNewSeminar.php" method="POST">
```

On *InsertNewSeminar.php* page

```
// Create short variable names
// Create short variable names
$SeminarDate = $_POST["SeminarDate"];
$SeminarTime = $_POST["SeminarTime"];
$Location = $_POST["Location"];
$SeminarTitle = $_POST["SeminarTitle"];
// Create SQL statement to INSERT new data
$SQLINSERT = "INSERT INTO SEMINAR ";
$SQLINSERT .= "VALUES('$SeminarDate',
    '$SeminarTime', '$Location', '$SeminarTitle')";
```

SQL Injection Attacks

- SQL Injection Attacks are attempts to issue SQL commands through a Web interface such as a Web form.
- Consider a Web form that asks for a user's new phone number:

```
// Create SQL statement
$varSQL = "UPDATE CUSTOMER SET PHONE = '$NewPhone' ";
$varSQL .= "WHERE CustomerID = '$CustomerID'";
// Execute SQL statement
  $RecordSet = odbc_exec($Conn, $varSQL);
```

- If the value of NewPhone is not checked, then we may get input values such as:

```
678-345-1234; DELETE FROM CUSTOMER;
```


XML Introduction

- Database processing and document processing need each other.
  - **Database processing** needs document processing for expressing database views.
  - **Document processing** needs database processing for storing and manipulating data.
- As Internet usage increases, organizations want to make their Web pages more functional by displaying and updating data from organizational databases.

XML

- **XML**, or **Extensible Markup Language**, was developed in the early 1990s.
  - XML is a subset of **SGML** or **Standard Generalized Markup Language.**
- Today XML is a hybrid of document processing and database processing.
  - It provides a standardized yet customizable way to describe the content of documents.
  - XML documents can automatically be generated from database data and vice versa.
- **SOAP** is an XML-based standard protocol for sending messages of any type, using any protocol over the Internet.

XML (Cont'd)

- XML is used for describing, representing and materializing database views.
- XML is better than HTML because:
  - It provides a clear separation between document structure, content and materialization.
  - It is standardized but allows for extension by developers.
  - XML tags accurately represent the semantics of their data.

SQL for XML Processing

SQL query with **FOR XML** clause

The query results are one cell containing the entire XML output—click on the results to display them in full

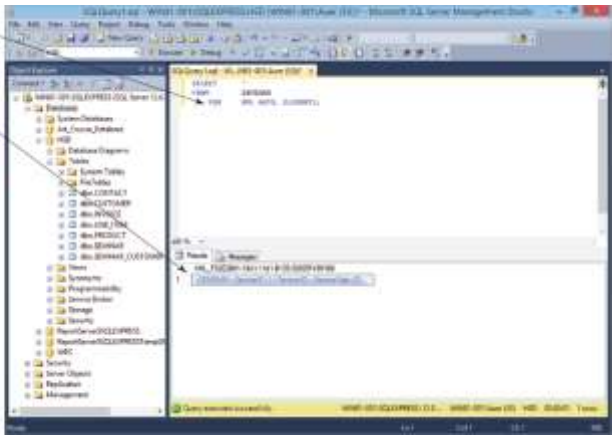


Figure 7-28: An SQL FOR XML Query

Results of SQL for XML Processing

The query results are now displayed in full—note the XML tags and data content

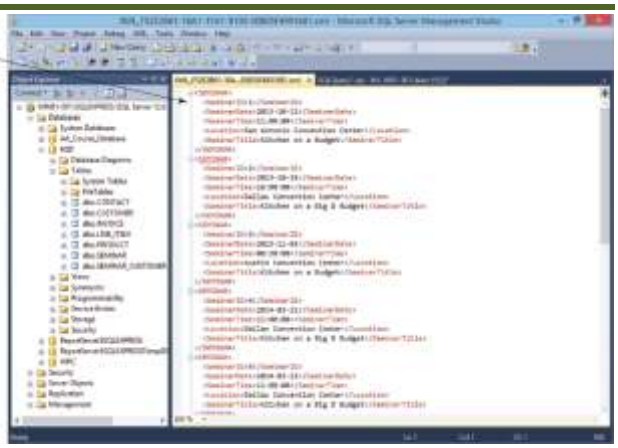


Figure 7-29: Results of the SQL FOR XML Query

XLM Web Services

- **XML Web Services** allow application functionality on one Web server to be shared and incorporated into Web applications on other Web servers.

DAVID M. KROENKE and DAVID J. AUER

DATABASE CONCEPTS, 7th Edition

End of Presentation on Chapter Seven

Database Processing Applications