Understanding Databases

Lesson 6

Objective Domain Matrix

Skills/Concepts	MTA Exam Objectives
Understanding Relational Database Concepts	Understand relational database management systems (6.1)
Understanding Database Query Methods	Understand database query methods (6.2)
Understanding Database Connection Methods	Understand database connection methods (6.3)

Data Anomalies

BookId	BookName	CategoryId	CategoryName
1	Cooking Light	1001	Cooking
2	Prophecy	1002	Mystery & Thriller
3	Shift	1003	Business
4	The Confession	1002	Mystery & Thriller

- Insert Anomaly
  - You cannot insert new data because of unrelated dependency
- Delete Anomaly
  - Deleting one piece of data causes unintended loss of other data
- Update Anomaly
  - Updating a single data value requires multiple rows to be updated

Data Normalization

- The process of data normalization ensures that a database design is free of any problems that could lead to loss of data integrity.
- This lesson discusses three normal forms
  - First Normal Form
  - Second Normal Form
  - Third Normal Form

First Normal Form (1NF)

- In order for a table to be in the first normal form (1NF), none of the columns in the table should have multiple values in the same row of data.
- The following table is not in 1NF because the PhoneNumber column is storing multiple va

Id	FirstName	LastName	PhoneNumber
1	Jane	Doe	(503) 555-6874
2	John	Doe	(509) 555-7969, (509) 555-7970
3.	Howard	Steel	(604) 555-3392, (604) 555-3393

Second Normal Form (2NF)

- For a table to be in second normal form (2NF), it must first meet the requirements for 1NF.
- 2NF also requires that all non-key columns are functionally dependent on the entire primary key.
- The following table violates the 2NF because the non-key columns are functionally dependent on only part of the primary key.

OrderId	CustomerId	OrderDate	CustomerName
101	1	10/1/2010	Jane Doe
102	2	10/5/2010	John Doe
103	1	10/4/2010	Jane Doe

Third Normal Form (3NF)

- For a table to be in third normal form (3NF), it must first meet the requirements for 2NF.
- 3NF also requires that that there is no functional dependency between non-key attributes.
- The following table violates the 3NF because the non-key columns are functionally dependent on only part of the primary key.

ItemId	SupplierId	ReorderFax
101	100	(514) 555-2955
102	11	(514) 555-9022
103	525	(313) 555-5735

Structured Query Language (SQL)

- SQL is the language used by most database systems to manage the information in their databases.
- SQL is declarative in nature -- you tell the database what needs to done, and it's the database's job to figure out how to do it.
- There are two main ways to submit T-SQL to SQL Server:
  - Ad-hoc SQL statements
  - Stored procedures

SQL Queries

- **SELECT**, **INSERT**, **UPDATE**, and **DELETE** statements are the four main types of SQL statements used to manipulate SQL Server data:
  - **SELECT** statement: retrieves data
  - **INSERT** statement: adds new data
  - **UPDATE** statement: modifies data
  - **DELETE** statement: deletes data

Running SQL Queries

- There are many way to communicate with SQL Server in order to run database queries:
  - Visual Studio Integrated Development Environment
  - C# application
  - SQL Query Analyzer
  - osql command prompt utility

Selecting Data

- The **SELECT** statement is used to retrieve data from one or more database tables.

SELECT *list_of_fields*
FROM *list_of_tables*
WHERE *where_clause*
GROUP BY *group_by_clause*
HAVING *having_clause*
ORDER BY *order_by_clause*

SELECT Examples

- The following **SELECT** statement matches each order with corresponding customer:

```
SELECT OrderID, Customers.CustomerId,
ContactName
FROM Orders INNER JOIN Customers
ON Orders.CustomerId = Customers.CustomerId
```

- The following **SELECT** statement gets all orders shipped to Canada:

```
SELECT *
FROM Orders
WHERE ShipCountry = 'Canada'
```

Updating Data

- The **UPDATE** statement is used to update information in database tables.
- The following statement a specific customer's contact name:

```
UPDATE Customers
SET ContactName = 'Maria Anderson'
WHERE CustomerId = 'ALFKI'
```

Inserting Data

- The **INSERT** statement is used to add one or more rows to a database table.
- The following statement inserts a new record to the Order Details table:

```
INSERT INTO [Order Details]
(OrderId, ProductId, UnitPrice, Quantity, Discount)
VALUES (10248, 2, 19.00, 2, 0)
```

Deleting Data

- The **DELETE** statement is used to remove information from database tables.
- The following statement deletes a record from the Customers table:

```
DELETE FROM Customers
WHERE CustomerId = 'ALFKI'
```

Stored Procedures

- A stored procedure is a set of SQL statements that is stored in a database.
- Stored procedures benefits:
  - You can use them to save complex SQL statements for future execution.
  - SQL Server compiles stored procedures so that they run faster than ad hoc queries.

Creating Stored Procedures

- You use T-SQL's **CREATE PROCEDURE** statement to create a stored procedure.
- The following stored procedure, when executed, returns all customers from France:

```
CREATE PROCEDURE GetCustomersFromFrance
AS
 SELECT * FROM Customers
 Where Country = 'France'
RETURN
```

Parameterized Stored Procedures

- The following stored procedure, when executed, returns total sales for a given customer:

```
CREATE PROCEDURE dbo.GetCustomerSales
  (
  @CustomerId char(5),
  @TotalSales money OUTPUT
  )
AS
  SELECT @TotalSales = SUM(Quantity * UnitPrice)
  FROM (Customers INNER JOIN Orders
  ON Customers.CustomerId = Orders.CustomerId)
  INNER JOIN [Order Details]
  ON Orders.OrderId = [Order Details].OrderId
  WHERE Customers.CustomerId = @CustomerId
RETURN
```

Working with Flat Files

- The data in a flat file can be plain text or binary. These files are called “flat files” to distinguish them from more structured forms of storage, such as relational databases and XML files.
- The **StreamReader** and **StreamWriter** classes allows you to manipulate text files.
- The **BinaryReader** and **BinaryWriter** classes allows you to manipulate binary files.

Extensible Markup Language (XML)

- XML is a text-based format for representing structured data.

```
<?xml version="1.0" encoding="utf-8"?>
<!--Customer List-->
<Customers>
  <Customer Id="ALFKI">
    <CompanyName>Alfreds Futterkiste</CompanyName>
    <Phone>030-0074321</Phone>
  </Customer>
  <Customer Id="EASTC">
    <CompanyName>Eastern Connection</CompanyName>
    <Phone>(171) 555-0297</Phone>
  </Customer>
</Customers>
```

Working with XML

- The classes that work with XML data are organized in the System.Xml namespace:
  - **XmlReader** and **XmlWriter**: These classes provide a fast, non-cached, forward-only way to read or write XML data.
  - **XmlDocument**: This class is an in-memory representation of XML data and allows navigation and editing of the XML document.

Working with DataSet

- A DataSet is an in-memory representation of relational data.
- A DataSet can have tables, relations, and data-integrity constraints such as unique constraints or foreign-key constraints.
- The DataAdapter acts as a bridge between the data source and the DataSet.
- The DataAdapter stores the data connection and data commands needed to connect to the data source. The DataAdapter also provides commands for retrieving data from the data source and commands for updating the data source with any changes.

Recap

- Relational Database Design
  - Entity Relationship Diagram
  - Data Normalization
- Database Query Methods
  - Working with SQL Queries
  - Working with Stored Procedures
- Database Connection Methods
  - Working with Flat Files, XML, DataSet