Unit 5

Physical Design
Key Concepts

• Create a database in SQL Server
• Create tables and attributes in SQL Server
• Define data types and nulls
• Set primary key and foreign keys
• Create a database diagram and relationships
• Enforce referential integrity
Terminology

- Physical design – based on the DBMS used
- Data types – constraint that determines the data that can be stored in a column
- Nulls – an unknown value
- Primary key - constraint that uniquely identifies each row in a table
- Foreign keys – constraint used to create relationships between tables
- Unicode - an extended character set that includes non-Latin character sets such as Russian or Japanese. It also includes the ASCII standard characters
- Referential integrity – enforcing the constraints of the relationships
Choosing a DBMS

- Compatibility with existing networks and systems
- Hardware and software requirements for the DBMS
- Features of the DBMS in relation to the data requirements
- Familiarity and expertise in the DBMS by IT staff
- Price and licensing
- Product reliability and support
Data Types

- Every column must have a data type declared
- Data type determines what kind of values can be stored in a column
- Numeric data types can store only numbers
- Character data types can store words, numbers, and other characters
- There are several ANSI data types but each DBMS has its own variations and additions
## Numeric Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
<th>Range/Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigint</td>
<td>8 bytes integer</td>
<td>(-2^{63} , (9,223,372,036,854,775,808) ) to (2^{63}-1 (9,223,372,036,854,775,807))</td>
</tr>
<tr>
<td>int</td>
<td>4 bytes</td>
<td>(-2^{31} , (-2,147,483,648) ) to (2^{31}-1 (2,147,483,647))</td>
</tr>
<tr>
<td>smallint</td>
<td>2 bytes</td>
<td>(-2^{15} , (-32,768) ) to (2^{15}-1 (32,767))</td>
</tr>
<tr>
<td>tinyint</td>
<td>1 byte</td>
<td>0 to 255</td>
</tr>
<tr>
<td>bit</td>
<td>1 bit</td>
<td>0, 1, or Null</td>
</tr>
<tr>
<td>decimal</td>
<td>User can set precision up to (10^{38})</td>
<td>decimal(10,2)</td>
</tr>
<tr>
<td>money</td>
<td>8 bytes</td>
<td>(-922,337,203,685,477.5808 ) to (922,337,203,685,477.5807)</td>
</tr>
<tr>
<td>smallmoney</td>
<td>4 bytes</td>
<td>(-214,748.3648 ) to (214,748.3647)</td>
</tr>
<tr>
<td>numeric</td>
<td>User can set precision up to (10^{38})</td>
<td>Same as decimal</td>
</tr>
<tr>
<td>float</td>
<td>Approximate numeric type, the number of bytes depends on number</td>
<td>(-1.79E+308 ) to (-2.23E-308, 0 ) and (2.23E-308 ) to (1.79E+308)</td>
</tr>
<tr>
<td>real</td>
<td>Also approximate, 4 bytes</td>
<td>(-3.40E + 38 ) to (-1.18E - 38, 0 ) and (1.18E - 38 ) to (3.40E + 38)</td>
</tr>
</tbody>
</table>
## Time Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
<th>Examples/Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>New in 2008, stores date values</td>
<td>January 1, 1 A.D. through December 31, 9999 A.D.</td>
</tr>
<tr>
<td>datetime2</td>
<td>New. Stores date and time and allows user to set precision in fractions of seconds</td>
<td>Same date range as above. Time range=00:00:00 through 23:59:59.9999999999</td>
</tr>
<tr>
<td>datetimeoffset</td>
<td>Date and time but with timezone awareness</td>
<td>same</td>
</tr>
<tr>
<td>smalldatetime</td>
<td>Smaller date time type</td>
<td>January 1, 1753, through December 31, 9999 00:00:00 through 23:59:59.997</td>
</tr>
<tr>
<td>time</td>
<td>New, you can set the precision in fractions of a second</td>
<td>00:00:00.000000000 through 23:59:59.9999999999</td>
</tr>
</tbody>
</table>
## Character Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>char</strong></td>
<td>Fixed-length ASCII text</td>
<td>“Jefferson” --max 255 characters</td>
</tr>
<tr>
<td><strong>text</strong></td>
<td>Text stores large blocks of text data. the text and ntext data types are deprecated, use varchar(MAX) or nvarchar(MAX)</td>
<td>2,147,483,647 bytes.</td>
</tr>
<tr>
<td><strong>varchar</strong></td>
<td>Variable-length ASCII</td>
<td>“Los Angeles”, Maximum 255 characters unless MAX (MAX allows 2^31-1 bytes)</td>
</tr>
<tr>
<td><strong>nchar</strong></td>
<td>Unicode fixed length</td>
<td>Uses Unicode UCS_2 character set</td>
</tr>
<tr>
<td><strong>ntext</strong></td>
<td>Unicode large block. Deprecated</td>
<td></td>
</tr>
<tr>
<td><strong>nvarchar</strong></td>
<td>Unicode variable length text</td>
<td></td>
</tr>
</tbody>
</table>
## Other Data Types

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>Variable-length binary data. The image data type is deprecated and will go away</td>
<td>$2^{31}-1$ bytes</td>
</tr>
<tr>
<td>binary</td>
<td>Fixed-length binary</td>
<td>1 to 8000 bytes</td>
</tr>
<tr>
<td>varbinary</td>
<td>Variable-length binary</td>
<td>1 to 8000 bytes unless you specify MAX, $2^{31}-1$ bytes</td>
</tr>
<tr>
<td>uniqueidentifier</td>
<td>Generates a unique identifier</td>
<td>6F9619FF-8B86-D011-B42D-00C04FC964FF</td>
</tr>
</tbody>
</table>
| xml       | Stores XML data as XML, can be validated against schema collections, queried with xquery | `<employee>`
             `<name>Sue Larson</name>`
`</employee>"
Nulls

- A null is an important concept in databases.
- A null represents the absence of a value, an unknown value.
- As such it is not equivalent to a 0 or an empty string “"”.
- Nulls are ignored by most aggregate functions such as count, sum, and average.
- You have a choice when designing a table to allow it to accept nulls or not.
Database Components

- Tables
- Views
- Diagrams
- Stored procedures
- Rules
- Defaults
- User-defined data types
- User-defined functions
- Users
- Roles
The database has two files:

- a primary data file that stores all the data objects and the data
- a log file that logs transactions as they occur.
Creating a Table in the GUI

- Right-click on the tables folder in the database
- Choose New Table
- Enter the fields and data types.
- Save and name the table

PRACTICE CREATING A Table
Creating a Table with SQL

CREATE TABLE Course
(
    CourseKey NCHAR(10) PRIMARY KEY,
    CourseName NVARCHAR(50) NOT NULL,
    CourseDescription NVARCHAR(200) NULL
)

Practice creating the Table
Database Diagram – Step 1

• To create a database diagram, right-click the database diagram folder under the database

• You will receive a message like the following:

   ![Microsoft SQL Server Management Studio message]

   This database does not have one or more of the support objects required to use database diagramming. Do you wish to create them?

   [Yes] [No]

PRACTICE THESE STEPS
Database Diagram – Step 2

• Click the “Yes” button. An Add Table dialog box appears:
Database Diagram – Results
Creating Relationships

- One way to create relationships in SQL Server is to select a primary key column in the parent table and drag the mouse to the matching field in the child table.

- This will bring up the Relationship dialog box.

PRACTICE THIS
Foreign Key Relationship

Next you can set properties for the relationship.
Referential Integrity

• Results from enforcing the foreign key constraint

• Enforcing it means that there can be no foreign key value in the child table that does not exist as a primary key value in the parent table

• Enforcing referential integrity prevents orphan records
## Enforcing Referential Integrity

<table>
<thead>
<tr>
<th>Action</th>
<th>Effect of Enforcing Referential Integrity</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT</td>
<td>You must enter data into the parent (primary key) table before you can enter data into a child (foreign key) table. Example: You must enter the customer information before entering the sale information.</td>
</tr>
</tbody>
</table>
| UPDATE | 1. You cannot change the primary key value for any record in the parent table without also changing the related foreign key. This creates a dilemma because both must be changed simultaneously. You can either suspend referential integrity while making the update or use Cascading Updates (see below).
   2. You can only update or change a foreign key in a child table to one that has a matching value in a parent or primary key table. |
| DELETE | You cannot delete a row in a primary key table unless all related records are first deleted in the foreign key table. Example: You can’t delete an order unless all the order details for that order are first deleted. |
Sample Data

• Once the tables are complete, it is a good idea to enter some sample data in order to test the database

• Make sure your sample data:
  – Are complete enough to test the data
  – Are varied enough to represent a variety of likely scenarios
  – Contain some exceptions and perhaps even some errors to test how your database handles them
Documentation

- The database is self-documenting
- System tables keep the “meta” information tables and other objects
- Best Practice - keep a separate data dictionary that describes all tables, columns, data types, and constraints
Summary

• We discussed how to create a database in the DBMS
• Create tables and attributes
• Define data types and nulls
• Set primary key and foreign keys
• Create a database diagram and relationships
• Enforce referential integrity
Lab 5

- Create the Westlake Research Hospital database using step by step instructions
- Apply what you just learned to the Grandfield college scenario