Introduction to Data Types

CREATE TABLE Transact-SQL Statement: **specify each column**

```
CREATE TABLE LineItems
(
    OrderID int NOT NULL
   ,ProductID int NOT NULL
   ,UnitPrice money NOT NULL
   ,Quantity smallint NOT NULL
   ,CONSTRAINT PK_LineItems
        PRIMARY KEY ([OrderID], [ProductID])
);
```

OrderID and **ProductID** are a combined Primary Key

Numeric Data Types

- Exact numeric data types:

Data type	Value range, between	Storage	
tinyint	0 and 255	1 byte	2^8
smallint	-32,768 and 32,767	2 bytes	2^{16}
int	-2,147,483,648 and 2,147,483,647	4 bytes	2^{32}
bigint	-9,223,372,036,854,775,808 and 9,223,372,036,854,775,807 (+/- 9 quintillion)	8 bytes	2^{64}
decimal	-10^{38} +1 through 10^{38} – 1 when maximum precision is used	5-17 bytes	

decimal(precision, scale) Example: decimal(8,2)

- Precision is the number of digits in a number (maximum 38)
- Scale is the number of digits to the right of the decimal point in a number

Character String Data Types

Each character requires one byte to store it

- Non-Unicode character string data types:
  - char (n)
  - varchar (n)
  - varchar (max)

Every value in a column will be the same length, as specified by the value (n)

Varchar stores only the required number of characters for each row

- Unicode character string types:
  - nchar (n)
  - nvarchar (n)

Unicode uses two bytes to store each character instead of one

Date and time data type

- Date and time data type:

Data type	Value range, between	Storage	
date	0001-01-01 and 9999-12-31	3 bytes	
time	00:00:00.0000000 and 23:59:59.9999999	5 bytes	
smalldatetime	Dates: 1900-01-01 and 2079-06-06 Times: 00:00:00 and 23:59:59	4 bytes	
datetime	Dates: January 1, 1753 and December 31, 9999 Times: 00:00:00 and 23:59:59.997	8 bytes	
datetime2	Dates: 0001-01-01 and 9999-12-31 Times: 00:00:00 and 23:59:59.9999999	8 bytes	

Keys and Constraints

- PRIMARY KEY constraints ensure uniqueness

```
CREATE TABLE Production.Categories
  (
    categoryid   INT  NOT NULL  IDENTITY,
    categoryname NVARCHAR(15)  NOT NULL,
    description  NVARCHAR(200) NOT NULL,
    CONSTRAINT PK_Categories PRIMARY KEY(categoryid)
  );
```

Keys and Constraints - FOREIGN KEY

```
CREATE TABLE Production.Categories
  (
     categoryid   INT  NOT NULL  IDENTITY,
     categoryname NVARCHAR(15)  NOT NULL,
     description  NVARCHAR(200) NOT NULL,
     CONSTRAINT PK_Categories PRIMARY KEY(categoryid)
  );

CREATE TABLE Production.Products
  (
     productid    INT  NOT NULL IDENTITY,
     productname  NVARCHAR(40) NOT NULL,
     supplierid   INT   NOT NULL,
     categoryid   INT   NOT NULL,
     unitprice    MONEY   NOT NULL,
     CONSTRAINT PK_Products PRIMARY KEY(productid),
     CONSTRAINT FK_Products_Categories FOREIGN KEY(categoryid)
          REFERENCES Production.Categories(categoryid)
  );
```

Keys and Constraints - UNIQUE constraints

- PRIMARY KEY constraint
- A table can have only one PK constraint
- A column with a PK constraint cannot include NULLs

- UNIQUE constraint
- A table can have multiple UNIQUE constraints
- A column with a UNIQUE constraint can include a single NULL value

UNIQUE constraints ensure uniqueness for non-primary key columns

Keys and Constraints

```
ALTER TABLE CustomerDemographics
        ADD CONSTRAINT DF_NumberChildren
                DEFAULT 'Unknown' FOR NumberOfChildren;
GO
```

- DEFAULT constraints provide a default value

Example

Customer ID	First Name	Last Name	Number of Children	City
1	Latasha	Navarro	2	Denver
2	Abby	Sai	NULL	Seattle

Potentially interpret the NULLs to mean 'zero children',
When in fact it means that the number of children is unknown

NULLs: potentially misleading or inaccurate query results

Keys and Constraints

```
ALTER TABLE Orders WITH NOCHECK ADD CONSTRAINT CK_OrderDate
        CHECK (OrderDate = GETDATE());
```

WITH NOCHECK option: Prevents the application of the constraint to existing values in the OrderDate column

- CHECK constraints check values against defined criteria

The Identity Property

- The IDENTITY property in a CREATE TABLE statement automatically generates integer values for a column
  - The seed value is the starting point of the numerical sequence

    IDENTITY (1, 1)
  - The increment value is the amount that the value increases by for each row

    IDENTITY (1, 1)

```
CREATE TABLE Orders
        (OrderID INT IDENTITY (1,1) NOT NULL
        ,CustomerID INT NOT NULL
        ,OrderDate DATETIME NOT NULL
        CONSTRAINT PK_OrderID PRIMARY KEY (OrderID));
```

The Identity Property

- The IDENTITY property in a CREATE TABLE statement automatically generates integer values for a column
  - The seed value is the starting point of the numerical sequence
  - The increment value is the amount that the value increases by for each row
- IDENTITY does not ensure uniqueness
  - Use a PRIMARY KEY or UNIQUE constraint
- Only one column per table can include the IDENTITY property

What Is a View?

- A view is a stored SELECT statement
  - Views mask database complexity for end users

CREATE VIEW statement

```
CREATE VIEW VW_CustomerOrders
AS
    SELECT C.CustomerID, Name, DateOfBirth, OrderID, OrderDate
    FROM Person.Customer AS C JOIN Sales.[Order] AS O
        ON C.CustomerID = O.CustomerID;
```

SELECT statement that references a view (as if it were a table)

```
SELECT * FROM VW_CustomerOrders;
```