Objectives

- SQL CREATE
- Examples
- Setting Primary Keys
  - References
- Foreign Keys
- Data types
- Easy way / Harder, but better way
SQL CREATE

- Syntax

```sql
CREATE TABLE <tablename>(
    <fieldname> <datatype>,
    <fieldname> <datatype>,
    <fieldname> <datatype>
);
```
Example 1

- Syntax

```sql
CREATE TABLE Employee(
    EmployeeID    char(15),
    LastName      char(30),
    FirstName     char(30)
);
```
Declaring a PRIMARY KEY

CREATE TABLE <tablename>(
    <fieldname> <datatype> PRIMARY KEY,
    <fieldname> <datatype>,
    <fieldname> <datatype>
);

SQL CREATE (cont.)
Example 2

Syntax

CREATE TABLE Employee(
EmployeeID char(15) PRIMARY KEY,
LastName varchar(30),
FirstName varchar(30)
);

Declares EmployeeID as the PRIMARY KEY of the Employee Table
Every table we create in this class will have a Primary Key.

- Remember that a PK is:
  - Unique
  - Not Null
Declaring a Composite PRIMARY KEY

CREATE TABLE <tablename>(
    <fieldname>  <datatype>,
    <fieldname>  <datatype>,
    <fieldname>  <datatype>,
    <fieldname>  <datatype>,
    PRIMARY KEY(<fieldname>, <fieldname>)
);
Example 3

Syntax

CREATE TABLE OrderDetails(
    OrderID           int,
    ProductID         int,
    UnitPrice         money,
    Quantity          int,
    PRIMARY KEY (OrderID, ProductID)
);
CREATE TABLE <tablename>(
    <fieldname> <datatype> PRIMARY KEY,
    <fieldname> <datatype> REFERENCES <tablename>(<fieldname>),
    <fieldname> <datatype>
);
Example 4

Syntax

CREATE TABLE OrderDetails(
    OrderID int REFERENCES Order(OrderID),
    ProductID int REFERENCES Product(ProductID),
    UnitPrice money,
    Quantity int,
    PRIMARY KEY(OrderID, ProductID)
);
Foreign Key

- A Foreign Key is a field in one table that references the Primary Key of another table.

- Use the keyword REFERENCES to create a Foreign Key
Order of Creation

- Tables without Foreign Keys must be created first in the database

- Then tables with Foreign Keys can be created

- Why?
CREATE TABLE <tablename>(
    <fieldname> <datatypename>
    CONSTRAINT <constraintname> PRIMARY KEY,
    <fieldname> <datatypename>
    CONSTRAINT <constraintname> REFERENCES <table1name>(<fieldname>),
    <fieldname> <datatypename>
    CONSTRAINT <constraintname> REFERENCES <table1name>(<fieldname>),
)
);
Example 5

Syntax

CREATE TABLE Order(
    OrderID int CONSTRAINT OrderPriKey PRIMARY KEY,
    CustomerID char(15) CONSTRAINT OrderCustID_FK REFERENCES Customer(CustomerID),
    EmployeeID int CONSTRAINT OrderEmpID_FK REFERENCES Employee(EmployeeID)
);

Referencing a composite Primary Key

CREATE TABLE <tablename>(
    <fieldname> <datatype>
    CONSTRAINT <constraintname> PRIMARY KEY,
    <fieldname> <datatype>,
    <fieldname> <datatype>,
    CONSTRAINT <constraintname>
    FOREIGN KEY(<fieldname>, <fieldname>)
    REFERENCES <tablename>(<fieldname>, <fieldname>)
);
Example 6

Syntax

CREATE TABLE CourseBook(
  ItemID    int REFERENCES Book(ItemID),
  CourseID  varchar(10),
  DepartmentID varchar(10),
CONSTRAINT CB_FK_cid_did
  FOREIGN KEY(CourseID, DepartmentID)
  REFERENCES Course(CourseID, DepartmentID),
CONSTRAINT CourseBookPriKey
  PRIMARY KEY(ItemID, CourseID, DepartmentID)
);
Syntax
CREATE TABLE Employee(
EmployeeID char(15) PRIMARY KEY,
Email varchar(50) UNIQUE,
LastName varchar(30),
FirstName varchar(30)
);

- Specifies that the value in Email must be unique, but it is not the primary key
Syntax

CREATE TABLE Employee(
    EmployeeID char(15) PRIMARY KEY,
    Email    varchar(50) UNIQUE,
    LastName varchar(30) NOT NULL,
    FirstName varchar(30) NOT NULL
);

- Specifies that the value in LastName and FirstName cannot be null. They must contain data.
ON DELETE CASCADE

Syntax

CREATE TABLE Order(
    OrderID   int    CONSTRAINT OrderPriKey PRIMARY KEY,
    CustomerID char(15) CONSTRAINT OrderCustID_FK
        REFERENCES Customer(CustomerID)
        ON DELETE CASCADE,
    EmployeeID int    CONSTRAINT OrderEmpID_FK
        REFERENCES Employee(EmployeeID)
        ON DELETE CASCADE
);
ON DELETE CASCADE

- Placed on a Foreign Key
- Used to enforce referential integrity

Whenever a row is deleted in the Customer table, any rows in Order that relate to that customer will also be deleted, hence, the deletion cascades.
ON UPDATE CASCADE

Syntax

```sql
CREATE TABLE Order(
    OrderID int CONSTRAINT OrderPriKey PRIMARY KEY,
    CustomerID char(15) CONSTRAINT OrderCustID_FK
        REFERENCES Customer(CustomerID)
        ON UPDATE CASCADE,
    EmployeeID int CONSTRAINT OrderEmpID_FK
        REFERENCES Employee(EmployeeID)
        ON UPDATE CASCADE
);
```
ON UPDATE CASCADE

- Placed on Foreign Keys
- Used to enforce referential integrity

Whenever a row is updated in the employee table, any rows in Order that relate to that employee will also be updated, hence, the update cascades.
Data Types

- In SQL Server:
  - int
  - datetime
  - decimal
  - char
  - money
  - varchar

- Many others, but these will satisfy the majority of your needs.
Harder, but better

- The way we have just learned is the somewhat harder way to create a database, but it is the better way.
- Writing a SQL script allows you to port that script to any DBMS
- Without that script, you will be forced to recreate the database, possibly by remembering this easier way…
Easier Way

- Look at Enterprise Manager
  - Create tables
  - Return all rows
  - Enter data
  - Create a diagram
  - Select data in Query Analyzer
SQL DROP

- DROP will remove a table from the database.
- If you want to completely remove a table and all of its contents, use DROP.
- This is often useful when you are finished debugging/developing and want to erase the database and start with fresh tables.

- DROP TABLE <tablename>;
- DROP TABLE Employee;