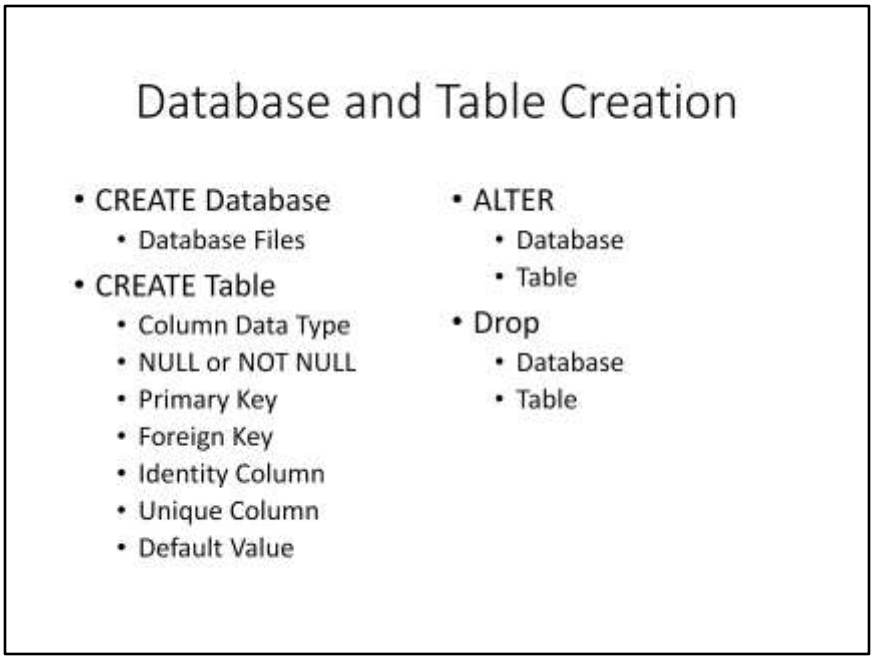
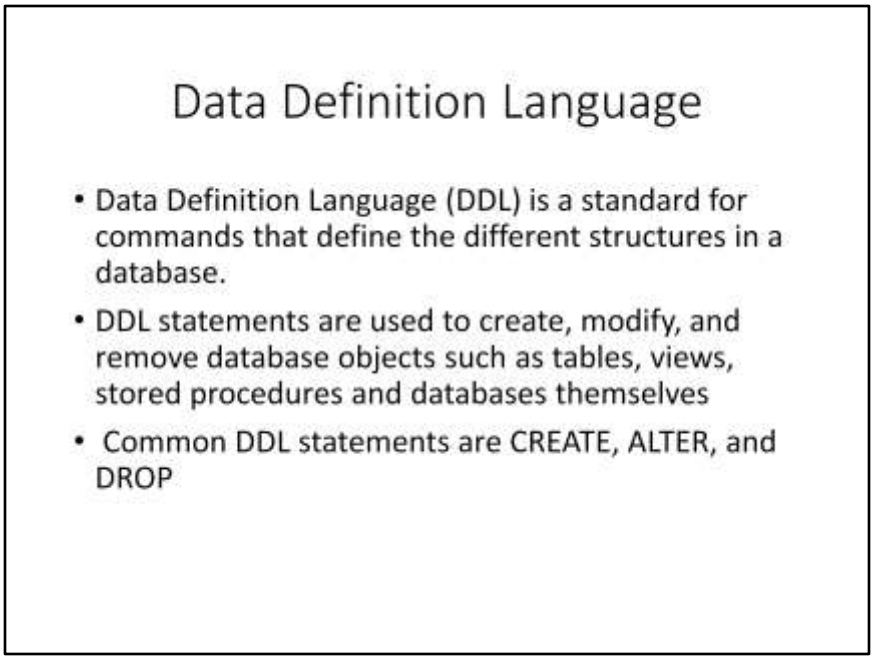


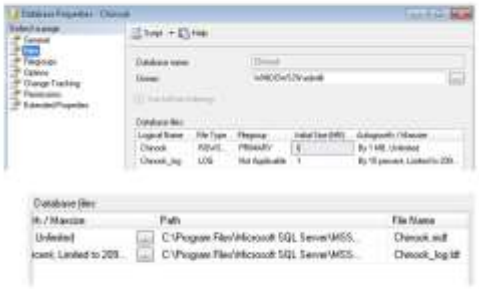
This presentation will cover SQL Server's data definition language. I will talk about the CREATE, ALTER, and DROP statements, and how you can use them to build a database and the tables contained therein.



The data definition language contains the commands used to create, modify and remove objects in a database. It differs from the data manipulation language in that it doesn't deal with the data itself, but rather with the objects associated within a database. The most common commands in the data definition language are the CREATE, ALTER and DROP statements. DDL is what you use to build a database and its attending objects.

Database Files

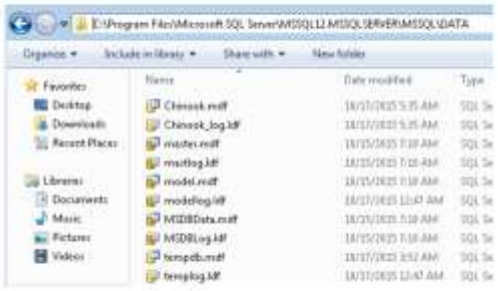
- SQL Server databases usually consist of 2 files by default. The Data file and the Log file
- The Data file contains the data and objects such as tables and views
- The Log file contains recent transactions and it needed for rollback and backup purposes
- Information in the Log file will eventually be pushed into the Data file.



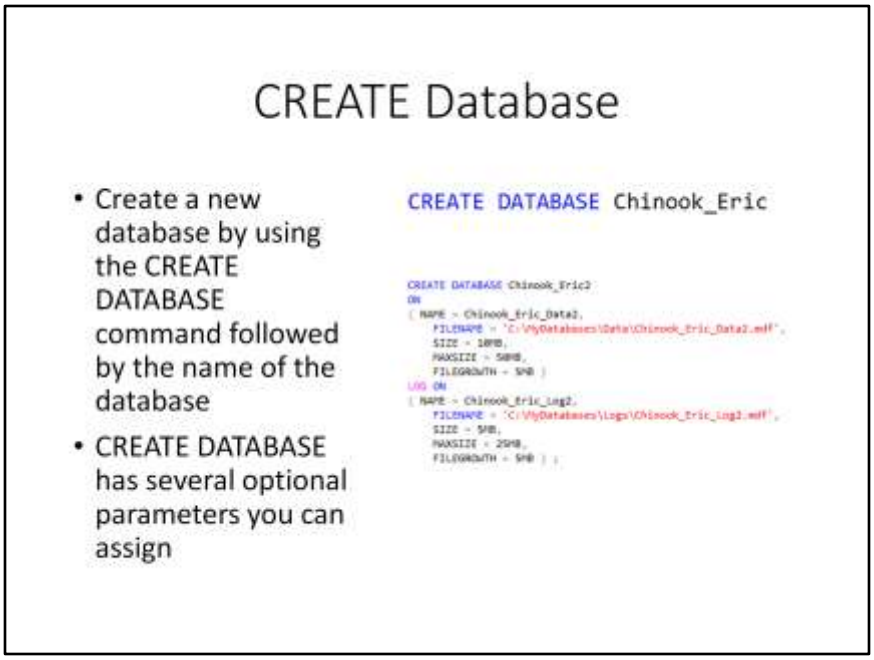
All SQL Server database information is stored in two types of files, the Data file and the Log file. By default there is only one of each file type per database, but it is possible to add more. The Data file is used to store all of the data and objects that are present in your database. The Log file contains the recent transaction history for your database. The Log file is needed when you have to roll back a transaction as well as in certain backup and restore scenarios. SQL Server occasionally issues Checkpoints which moves data from the Log file to the Data file. The log file is then truncated. Management of the Data and Log files is an advanced topic beyond the scope of this class.

Database File Location

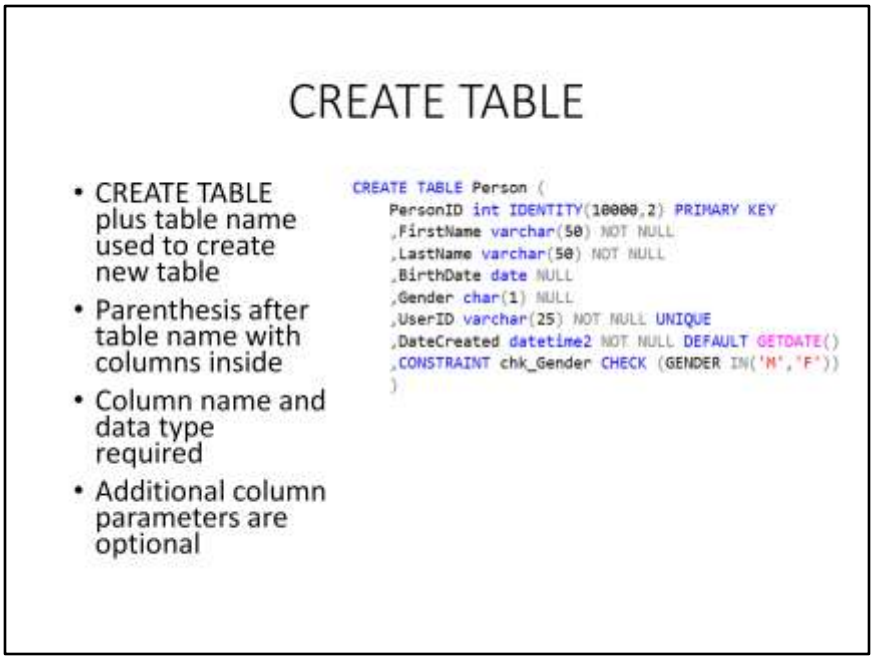
- Default location for SQL Server database files is C:\Program Files \Microsoft SQL Server \ [Server Version] \MSSQL\DATA
- The default location can be changed



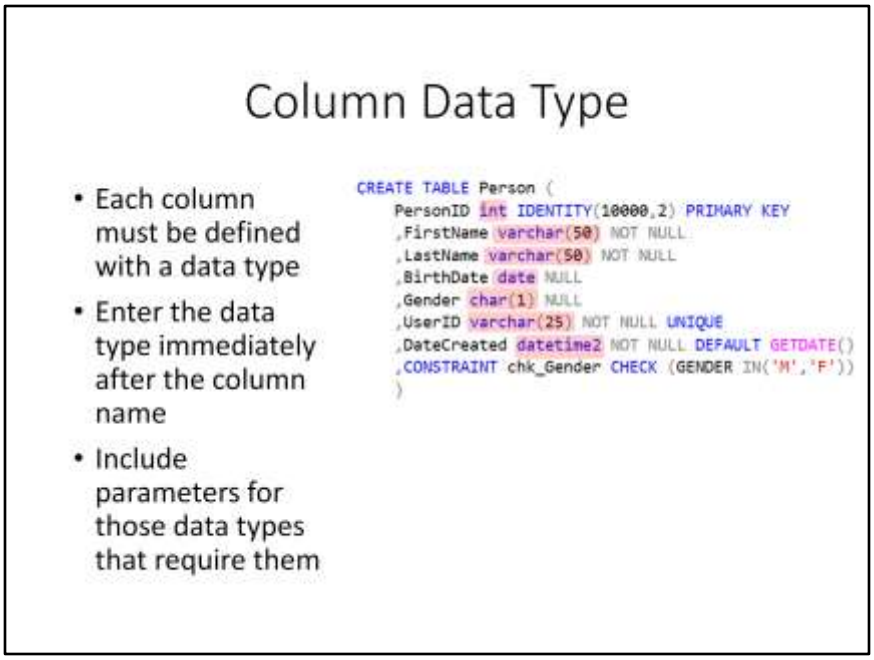
By default the Data and Log files are stored in the same location where you installed SQL Server. This is usually on the C drive under Program Files and Microsoft SQL Server. The default location for Data and Log files can be changed, and it is often recommended that you do so with larger databases for performance reasons.



Creating a new database can be very simple. Going with the defaults you can create a database by simply typing the CREATE DATABASE keywords followed by the name you want to give the new database. There are multiple parameters available if you want to be more specific with your database creation. For example you can choose the locations of your data and log files. You can also choose the initial and maximum sizes of your database, as well as in what increments to grow the database by.



The CREATE TABLE keywords are used to create a new table in your database. The name of the table follows the keywords. After the table name you must provide a list of the columns enclosed in parenthesis. The column names are required as are their associated data types. Additional column options such as constraints, defaults, primary keys and nullability may be included.



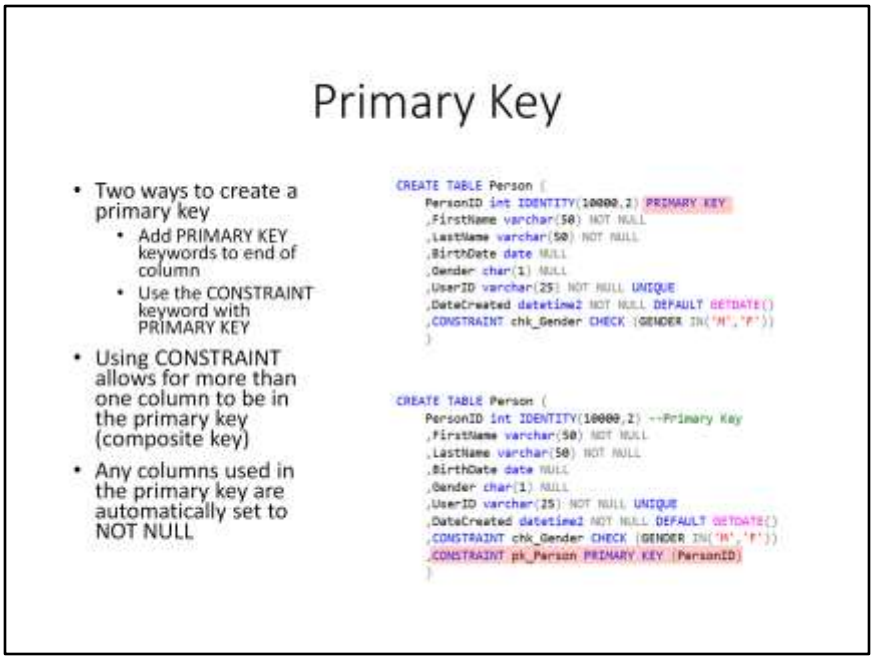
When creating a new table it is mandatory that you assign data types to each column you include in the table definition. The data type comes immediately after the name you give your column, and before anything else related to the column. Examples of data types are highlighted on this slide. Don't forget to include any parameters for your data types if they need them.

NULL or NOT NULL

- Columns can be flagged to allow (NULL) or prevent (NOT NULL) null values
- Assigning nullability is optional
- If nullability is not assigned then the column allows NULL values by default

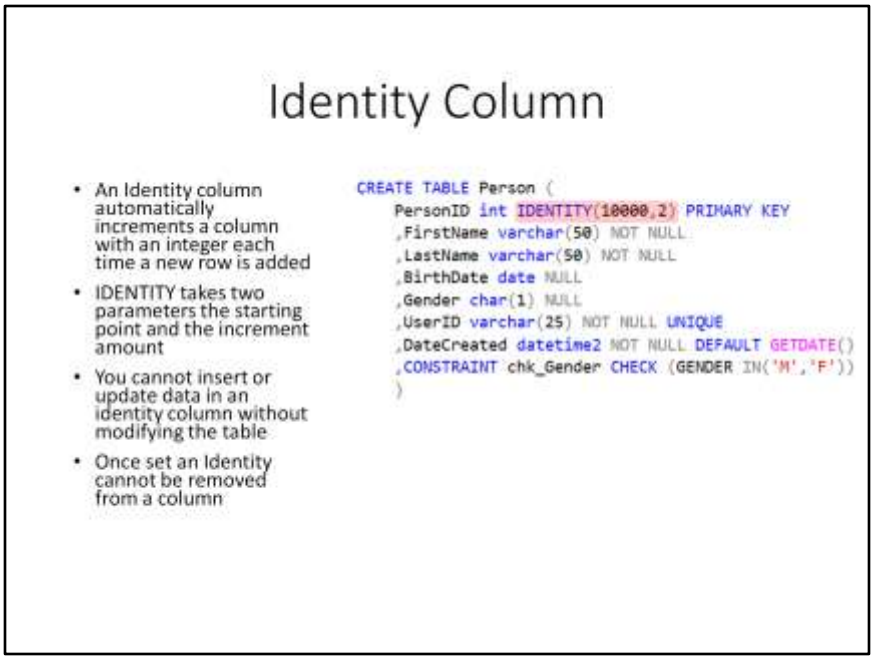
```
CREATE TABLE Person (
    PersonID int IDENTITY(10000,2) PRIMARY KEY
    ,FirstName varchar(50) NOT NULL
    ,LastName varchar(50) NOT NULL
    ,BirthDate date NULL
    ,Gender char(1) NULL
    ,UserID varchar(25) NOT NULL UNIQUE
    ,DateCreated datetime2 NOT NULL DEFAULT GETDATE()
    ,CONSTRAINT chk_Gender CHECK (GENDER IN('M','F'))
    )
```

All columns in a new table can be flagged as NULL or NOT NULL. If a column is flagged as NOT NULL then you will not be allowed to execute an INSERT on the table unless that column, and any others with NOT NULL are provided a value. If you don't specify a NULL flag then a column is deemed NULL by default. Note that some constraints such as the Primary Key will automatically convert a column to NOT NULL.



The Primary Key is used to uniquely identify a record in a table. There are two syntax options for defining the Primary Key on a table. One is by adding the PRIMARY KEY keywords to the end of the column. The other is to add a PRIMARY KEY CONSTRAINT to the table. The first option is quicker but your are limited to one column in your Primary Key. The constraint option has the advantage of allowing you to specify more than one column for your primary key. Each additional column needs to be separated by a comma. You are also able to choose the name of your Primary Key with the constraint option.

The Foreign Key is used to identify relations between tables. The Foreign Key in one table will reference the Primary in another. Like the Primary Key there are two syntax options for the Foreign Key. The first option goes on the same line as the column to be identified as the foreign key. You add the key words FOREIGN KEY REFERENCES plus the Primary Key table name followed by the column name in parenthesis. The constraint option requires you identify the Foreign Key column as well as the Primary Key column. You are allowed you have multiple columns in the CONSTRAINT FOREIGN KEY syntax. The foreign key data type must have the same data type as the primary key data type.



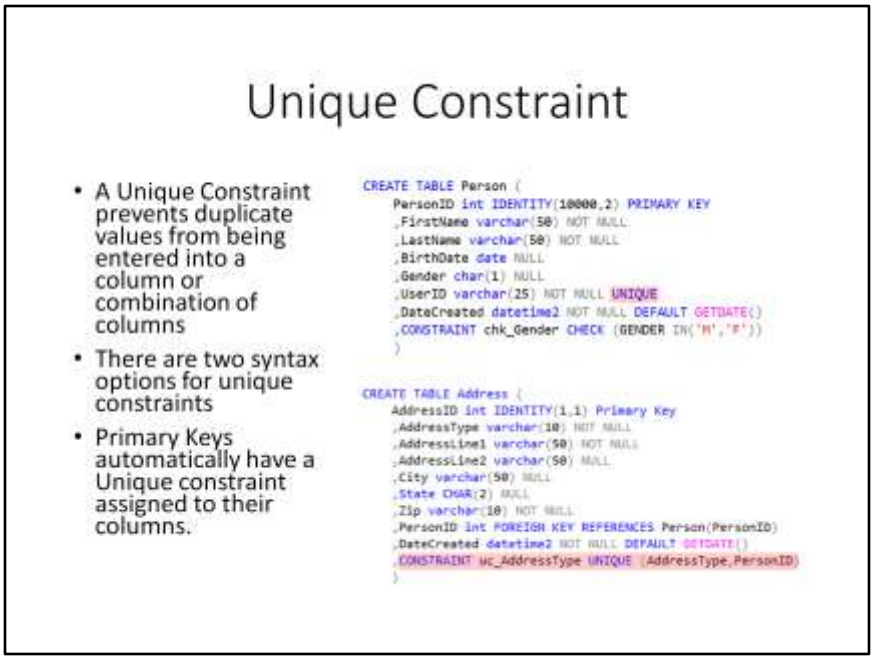
An identity column inserts an integer value automatically into itself when a new row is added to a table. The identity increments by a fixed value each time a new row is added so you will never have the same value twice. It is not possible to manually insert or update an identity column without an explicit override. The syntax for creating an identity column is adding the IDENTITY keyword plus its two parameters after the column data type. The first parameter tells SQL Server what integer to start with, and the second says how many units to increment by. In the example the highlighted text shows that the first record will have a value of 10,000. The second 10,002. The third 10,004 and so on. Once you create an identity column, the only way to remove the identity is to delete the entire column.

Column Default Value

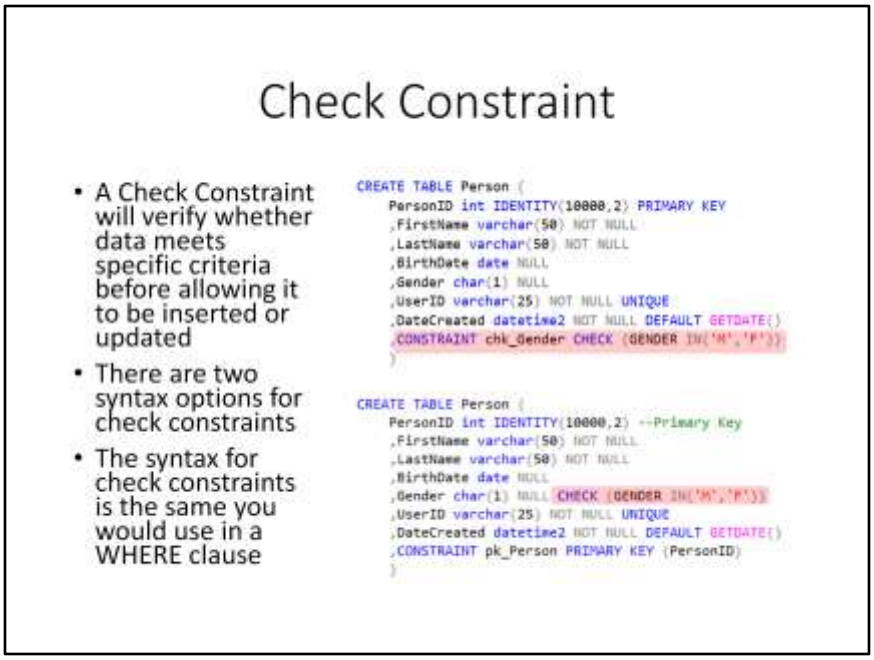
- A column can be assigned a default value
- If an INSERT statement doesn't include a value for the column, its default value will be used

```
CREATE TABLE Person (
    PersonID int IDENTITY(10000,2) PRIMARY KEY
    ,FirstName varchar(50) NOT NULL
    ,LastName varchar(50) NOT NULL
    ,BirthDate date NULL
    ,Gender char(1) NULL
    ,UserID varchar(25) NOT NULL UNIQUE
    ,DateCreated datetime2 NOT NULL DEFAULT GETDATE()
    ,CONSTRAINT chk_Gender CHECK (GENDER IN('M','F'))
    )
```

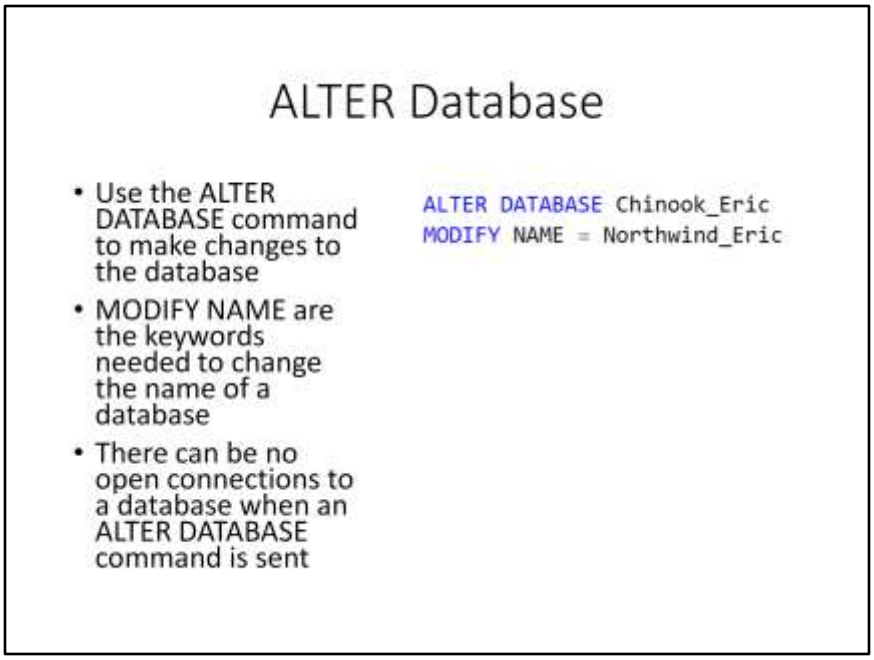
You can assign a default value to a column. When a row is added to a table and a value isn't specified for the column, then the default value will be entered in its place. The syntax for a default value is the keyword DEFAULT followed by the value you wish to enter. In the example the DateCreated column has a default of GetDate(). This means the current date and time will be entered into the column each time a new row is added.



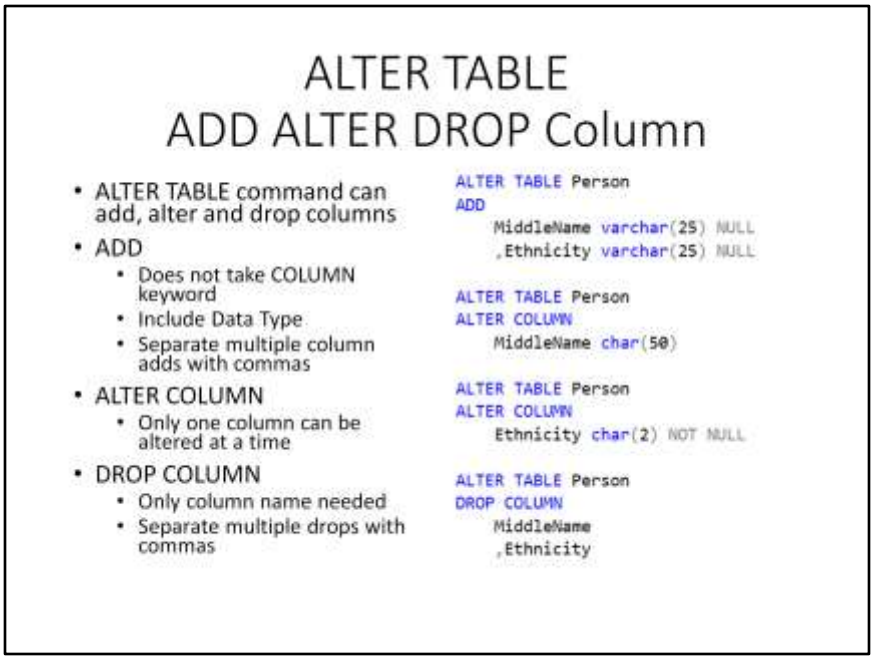
A unique constraint tells SQL Server that duplicate values are not allowed for a column. There are two syntax options for the unique constraint. The first is simply adding the keyword UNIQUE to the end of the column you want to be unique. The second consists of the CONSTRAINT keyword followed by the constraint name, the UNIQUE keyword and the column or columns that must be unique. In the highlighted portion of the second example I placed a unique constraint on AddressType and PersonID. This means it is not possible for a person to have the same address type more than once. One difference between unique and primary key is that unique allows for a NULL value while primary key does not.



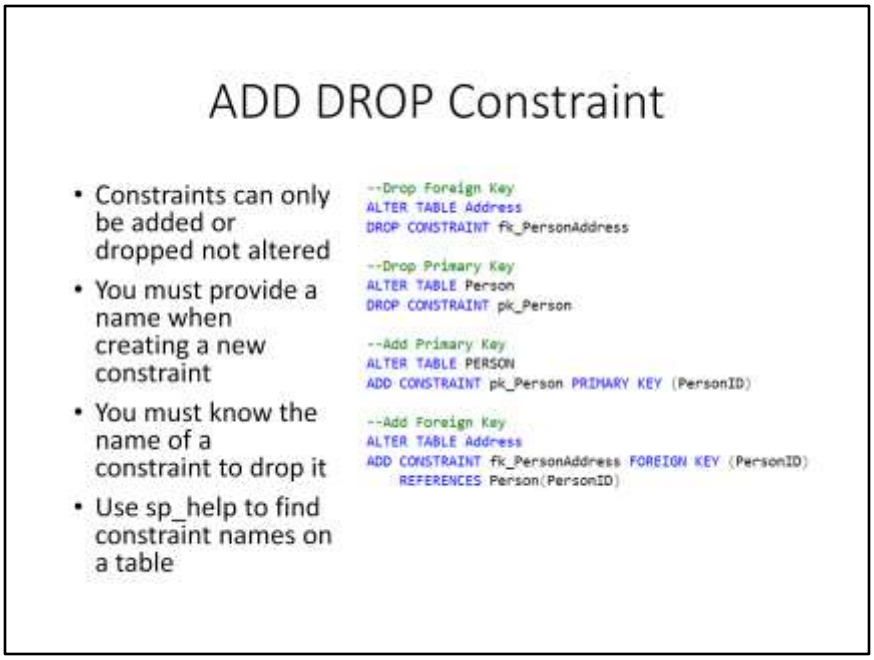
A check constraint will test the validity of a value before inserting or updating it to a column. If the value doesn't meet the check constraint requirements then an error will be thrown and the entire insert or update statement will be rolled back. There are two syntax options for the check constraint. The first is adding the CHECK keyword to the end of the column followed a condition in parenthesis. The condition can be similar to anything you would find in a WHERE clause. The second option uses the CONSTRAINT keyword followed by the constraint name, the CHECK keyword and the condition. In both examples the condition is checking whether the Gender value is equal to an M or an F. A NULL value is acceptable because the Gender column accepts nulls.



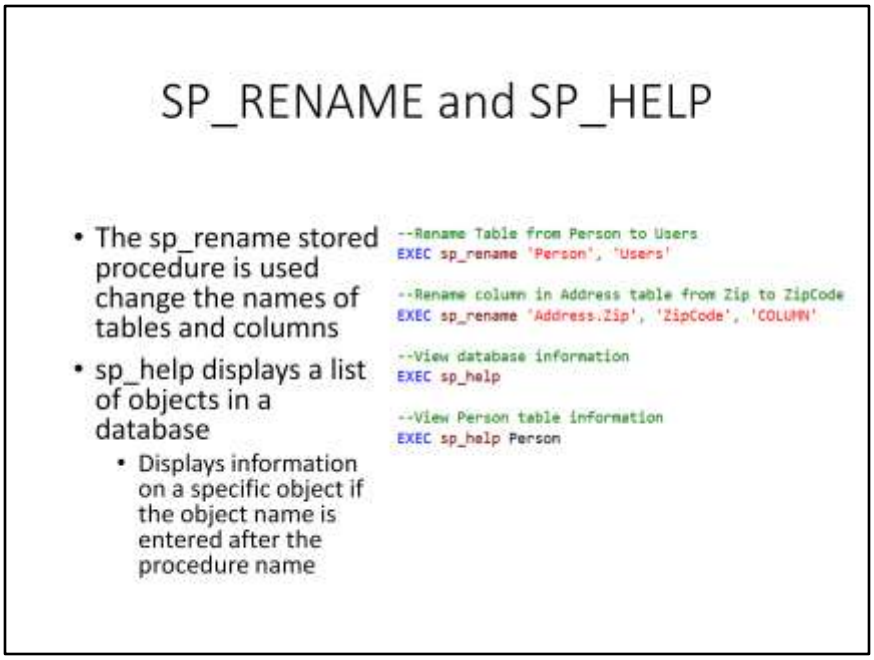
You can make modifications to a database by using the ALTER DATABASE command. In the example I am using alter database with the MODIFY NAME keywords to change the name of the database from Chinook_Eric to Northwind_Eric. An alter database command can only succeed against a database that has no connections open to it.



You can add commands to the ALTER TABLE statement that will allow you to add, alter or drop columns within the table. The ADD keyword is used when you wish to add columns to an existing table. In the example I added the columns MiddleName and Ethnicity to the Person table. Note that I still needed to include the data types, and each column is separated by a comma. The ALTER COLUMN keywords are used to modify a column in a table. Note that only one column can be modified per ALTER TABLE statement. In the example I changed the MiddleName data type to a char(50), and then I changed Ethnicity to a char(2) and set it to NOT NULL. The DROP COLUMN keywords are used to remove columns from a table. Only the column names are needed with each column separated by a comma. In the example I am removing the MiddleName and Ethnicity columns from the Person table.

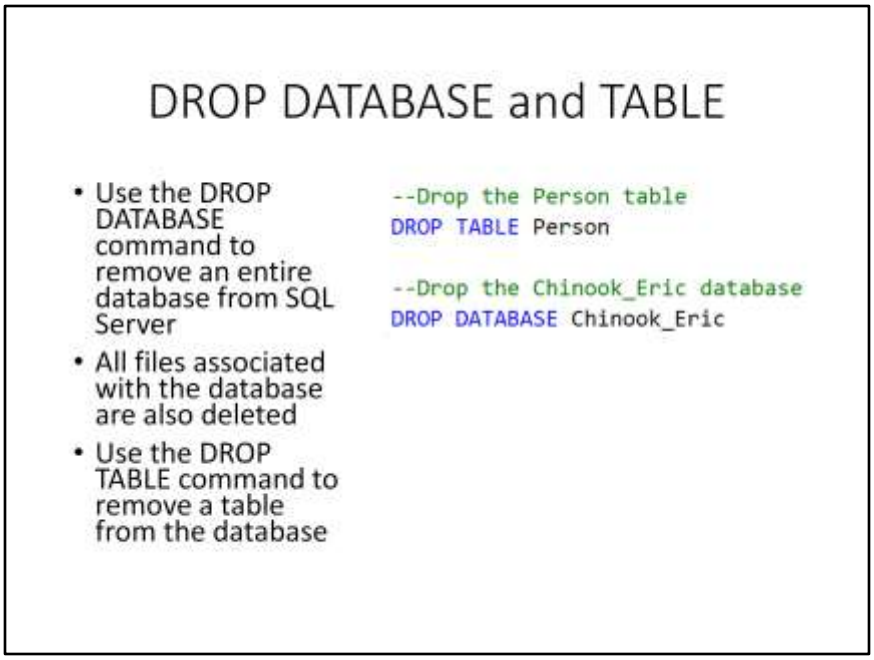


Constraints can be added or dropped from a table, but not modified. If you wish to change something in a constraint, you will have to drop it and then recreate it with the new criteria. It is required that you know or choose a name for your constraints when dropping or adding them to your table. You can look up constraint names by using the sp_help stored procedure. In the example I am dropping the foreign key from the Address table and the primary key from the Person table by using the DROP CONSTRAINT command. I am then recreating the primary and foreign keys using the ADD CONSTRAINT command.



In SQL Server the ALTER TABLE command does not have the ability to change the name of a table or a column. In order to achieve this you need to use the sp_rename stored procedure. The procedure takes up to 3 parameters. For a table rename the first parameter is the current table name and the second is the new table name. For a column rename the first parameter is the table and column name separated by a period. The second parameter is the new column name and the third parameter is the word “column”.

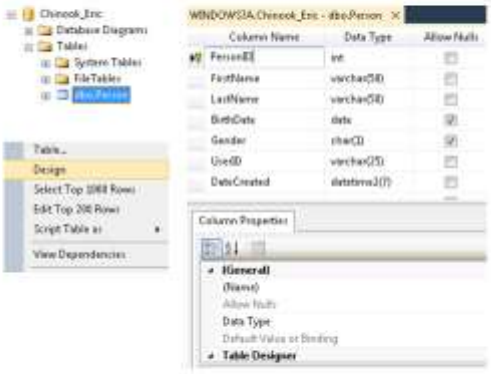
The sp_help stored procedure displays information about the database you’re in, or an object in the database. It takes 1 optional parameter. If you just execute sp_help it will return information on the database. If you execute sp_help with an object name, information on that object will be displayed.



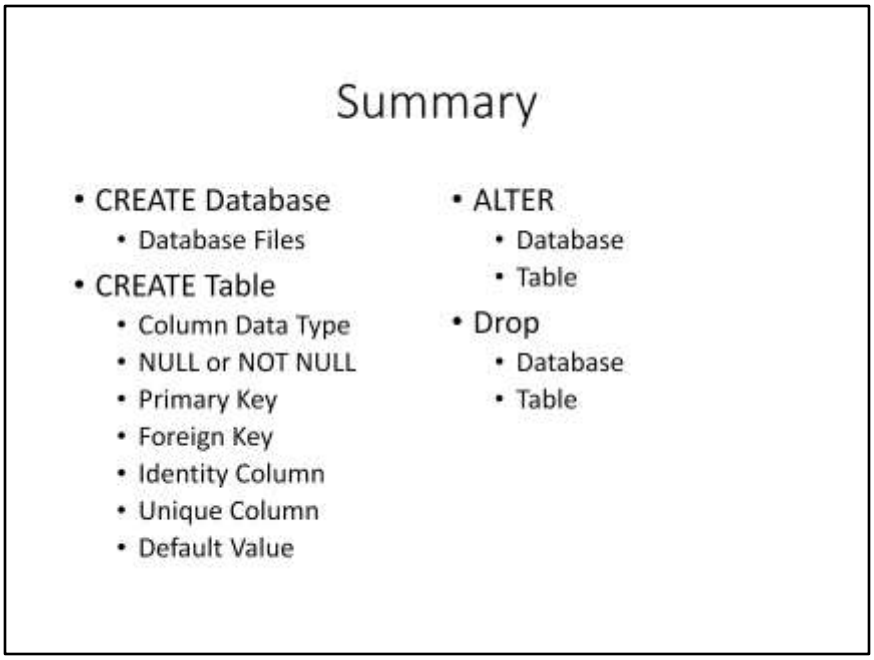
The DROP command is pretty straight forward. If you want to remove a table from your database, you use the DROP TABLE keywords followed by the table name. If you wish to remove an entire database you use the DROP DATABASE keywords followed by the database name. Note you cannot delete a database if there are any active connections to the database. This includes the connection you are executing the DROP DATABASE command from. It is best to operate from the master database when issuing drop database commands.

Using Management Studio

- Management Studio is an effective alternative for executing CREATE, ALTER and DROP commands
- Using SSMS can be particularly effective when you need to make numerous alterations to a table



SQL Server Management Studio is an effective tool for executing your data definition language statements. It provides a GUI interface that is more intuitive than writing code. It is particularly effective in executing multiple ALTER statements at one time. It is important as a SQL programmer that you know how to write DDL statements, but using SSMS to manage your data objects can be a viable alternative.



This concludes the presentation on database and table creation.