

The background of the slide features a collage of business and technology-related icons. On the left, there is a large, glowing database cylinder. In the center, a list of business processes is visible: Innovation, Branding, Solution, Marketing, Analysis, Idea, Success, and Management. To the right, a hand is shown drawing a lightbulb, with other icons like a bar chart, a network diagram, a puzzle piece, and a pie chart scattered around. The entire image has an orange tint.

# Intelligent Data Management with SQL Server

**Session: 6**

## *Creating Tables*

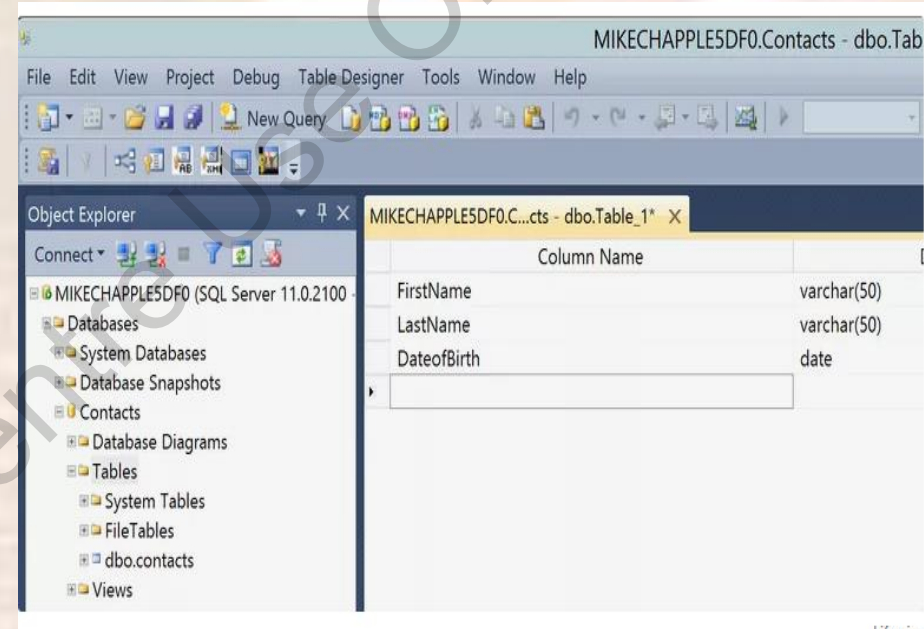
# Objectives

- List SQL Server 2019 data types
- Describe the procedure to create, modify, and drop tables in an SQL Server database
- Describe the procedure to add, modify, and drop columns in a table

For Aptech Centre Use Only

# Introduction

- One of the most important types of database objects in SQL Server 2019 is a table.
- Tables in SQL Server 2019 contain data in the form of rows and columns.
- Each column may have data of a specific type and size.



# Advanced Data Types

Name	Description
hierarchyid	It is a system data type with variable length. You can use it to represent a position in a hierarchy.
geometry	It is a spatial data type, implemented as a Common Language Runtime (CLR) data type in SQL Server. It represents data in a Euclidean (flat) coordinate system. SQL Server supports a set of methods for this data type. The geometry type is predefined and available in each database. You can create table columns of type geometry and operate on geometry data similar to how you do on other CLR types.
geography	It is a spatial type for storing ellipsoidal (round-earth) data, such as GPS latitude and longitude coordinates. SQL Server supports a set of methods for the geography spatial data type. This type too is predefined and available in each database. You can create table columns of type geography.
xml	It is a special data type for storing XML data in SQL Server tables.
cursor	It is a data type for variables or stored procedure OUTPUT parameters that contain a reference to a cursor.
table	It is a special data type useful for storing result set temporarily in a table-valued function. You can use data from this for processing later. It can be used in functions, stored procedures, and batches
rowversion	It returns automatically generated, unique binary numbers within a database.

## Advanced Data Types in SQL Server

# Creating, Modifying, and Dropping Tables

Most tables have a primary key, made up of one or more columns of the table

A primary key is always unique

The Database Engine will enforce the restriction that any primary key value cannot be repeated in the table

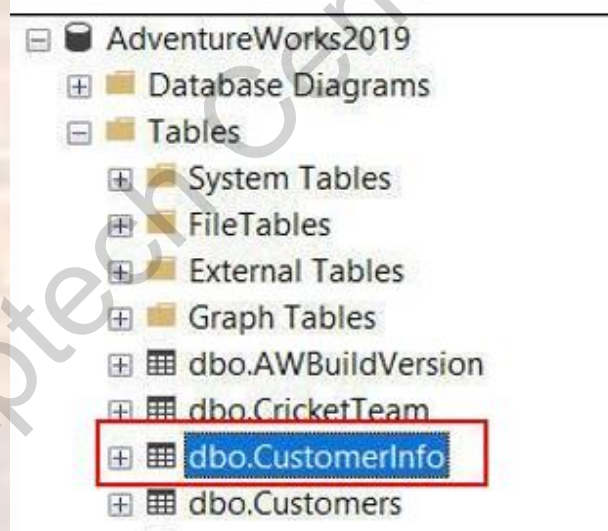
Thus, the primary key can be used to identify each record uniquely



# Creating Tables

The CREATE TABLE statement is used to create tables in SQL Server 2019. A simple basic syntax is as follows:

```
CREATE TABLE [database_name. [schema_name].| schema_name.]table_name  
    ([<column_name>] [data_type] Null/Not Null,)  
    ON [filegroup | "default"]
```



Creating a Table

# Modifying Tables

The ALTER TABLE statement is used to modify a table definition by altering, adding, or dropping columns and constraints, reassigning partitions, or disabling or enabling constraints and triggers.

## Syntax:

```
ALTER TABLE [[database_name].[schema_name].| schema_name.]table_name  
    ALTER COLUMN ([<column_name>] [data_type] Null/NotNull,);  
| ADD ([<column_name>] [data_type] Null/NotNull,);  
| DROP COLUMN ([<column_name>];
```

# Dropping Tables

The DROP TABLE statement removes a table definition, its data, and all associated objects such as indexes, triggers, constraints, and permission specifications for that table.

## Syntax:

```
DROPTABLE <Table Name>
```



# Data Modification Statements 1-2

The statements used for modifying data are INSERT, UPDATE, and DELETE statements.

These are explained as follows:

- **INSERT Statement** - The INSERT statement adds a new row to a table.

```
INSERT [INTO] <Table_Name> VALUES <values>
```

Syntax for INSERT Statement

# Data Modification Statements 2-2

- **UPDATE Statement** - The UPDATE statement modifies the data in the table.

```
UPDATE <Table_Name>  
SET <Column_Name=Value>  
[WHERE <Searchcondition>]
```

Syntax for UPDATE Statement

- **DELETE Statement** - The DELETE statement removes rows from a table.

```
DELETE FROM <Table_Name> [WHERE <Search condition>]
```

Syntax for DELETE Statement

# Column Nullability

The nullability feature of a column determines whether rows in the table can contain a null value for that column. In SQL Server, a null value is not same as zero, blank, or a zero length character string (such as ' ')

For example:

A null value in Color column of Production.Product table of AdventureWorks2019 database does not mean that the product has no color; it just means that color for the product is unknown or has not been set.

Nullability of a column can be defined either when creating a table or modifying a table. The NULL keyword is used to indicate that null values are allowed in the column and NOT NULL is used to indicate that null values are not allowed.

# DEFAULT Definition 1-2

A DEFAULT definition can be given for the column to assign it as a default value if no value is given at the time of creation.

For example:

It is common to specify zero as the default for numeric columns or 'N/A' or 'Unknown' as the default for string columns when no value is specified.

Results		Messages	
	ProductID	Name	Price
1	111	Rivets	100.00

Demonstrating Use of DEFAULT

# DEFAULT Definition 2-2

Following cannot be created on columns with DEFAULT definitions:

A timestamp data type

An IDENTITY or ROWGUIDCOL  
property

An existing default definition  
or default object



# IDENTITY Property 1-3

- The IDENTITY property of SQL Server is used to create identifier columns that can contain auto-generated sequential values to uniquely identify each row within a table.

For example:

An identifier column could be created to generate unique student registration numbers automatically whenever new rows are inserted into the Students table.

- The identity number for the first row inserted into the table is called seed value.
- The increment, also called Identity Increment property, is added to the seed in order to generate further identity numbers in sequence.

# IDENTITY Property 2-3

A column having `IDENTITY` property must be defined using one of the following data types:

`decimal, int, numeric, smallint, bigint, or tinyint`

A column having `IDENTITY` property need not have a seed and increment value specified. If they are not specified, a default value of 1 will be used for both.

A table cannot have more than one column with `IDENTITY` property.

The identifier column in a table must not allow null values and must not contain a `DEFAULT` definition or object.

Columns defined with `IDENTITY` property cannot have their values updated.

The values can be explicitly inserted into the identity column of a table only if the `IDENTITY_INSERT` option is set `ON`. When `IDENTITY_INSERT` is `ON`, `INSERT` statements must supply a value.

Characteristics of the `IDENTITY` property

# IDENTITY Property 3-3

The advantage of identifier columns is that SQL Server can automatically provide key values, thus reducing costs that would have been incurred for extra storage and improving performance.

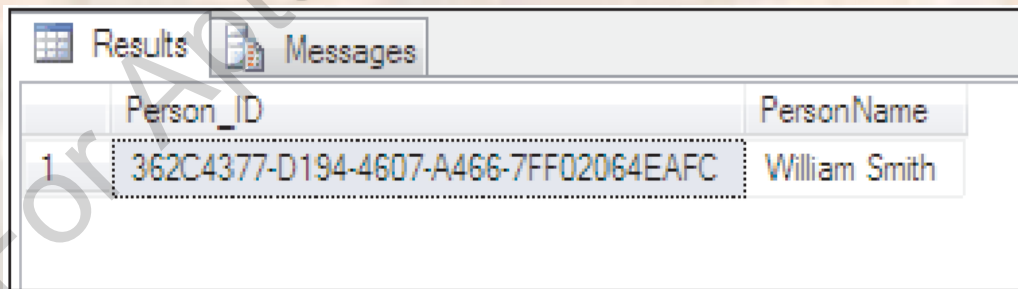
Results		Messages
	Person_ID	MobileNumber
1	500	983452201
2	501	993026654

IDENTITY Property Applied on Person\_ID Column

# Globally Unique Identifiers

In a networked environment, many tables may require to have a column consisting of a common globally unique value.

- Consider a scenario where data from multiple database systems such as banking databases must be consolidated at a single location.
- When the data from around the world is collated at the central site for consolidation and reporting, using globally unique values prevents customers in different countries from having the same bank account number or customer ID.



Person_ID	PersonName
1	William Smith

Unique Identifier

# Constraints 1-2

One of the important functions of SQL Server is to maintain and enforce data integrity.

- A constraint is a property assigned to a column or set of columns in a table to prevent certain types of inconsistent data values from being entered.
- Constraints are used to apply business logic rules and enforce data integrity.



# Constraints 2-2

Constraints can be categorized as column constraints and table constraints.

Column Constraint	Table Constraint
Is specified as part of a column definition and applies only to that column.	Can apply to more than one column in a table and is declared independently from a column definition. Table constraints must be used when more than one column is included in a constraint.

SQL Server supports the following types of constraints:



# PRIMARY KEY

- A table typically has a primary key comprising a single column or combination of columns to uniquely identify each row within the table.
- The PRIMARY KEY constraint is used to create a primary key and enforce integrity of the entity in the table.

```
Messages

(1 row affected)
Msg 2627, Level 14, State 1, Line 1
Violation of PRIMARY KEY constraint 'PK__EmpConta__7811348110185FA5'.
Cannot insert duplicate key in object 'dbo.EmpContactPhone'.
The duplicate key value is (101).
The statement has been terminated.
```

Output Error Message for Duplicate EMP\_ID

Results		Messages		
	EMP_ID	MobileNumber	ServiceProvider	LandlineNumber
1	101	983345674	Verizon	NULL

Output of the Successfully Executed First INSERT Statement

# UNIQUE

A UNIQUE constraint is used to ensure that only unique values are entered in a column or set of columns. It allows developers to make sure that no duplicate values are entered.

- Primary keys are implicitly unique.
- Unique key constraints enforce entity integrity because once the constraints are applied; no two rows in the table can have the same value for the columns.
- UNIQUE constraints allow null values.
- A single table can have more than one UNIQUE constraint.

```
Messages

(1 row affected)
Msg 2627, Level 14, State 1, Line 4
Violation of UNIQUE KEY constraint 'UQ__NewEmpCo__0A8C89724EBCC374'.
Cannot insert duplicate key in object 'dbo.NewEmpContactPhone'.
The duplicate key value is (<NULL>).
The statement has been terminated.
```

Output Error Message for Value Duplicate MobileNumber

Results		Messages		
	Person_ID	MobileNumber	ServiceProvider	LandlineNumber
1	111	983345674	Verizon	NULL

Successfully Inserted Row

# FOREIGN KEY

A foreign key in a table is a column that points to a primary key or unique column in a table. Foreign key constraints are used to enforce referential integrity.



Database Diagram Showing Foreign Key and Relationship Between Tables

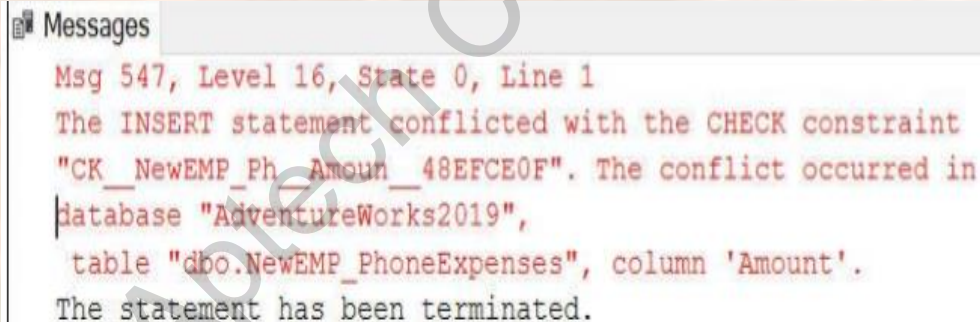
```
Results Messages
Msg 547, Level 16, State 0, Line 1
The INSERT statement conflicted with the
FOREIGN KEY constraint "FK_EmpPhoneE_Mobil_30E33A54".
The conflict occurred in database "AdventureWorks2019",
table "dbo.NewEmpContactPhone", column 'MobileNumber'.
The statement has been terminated.

(0 rows affected)
```

Output Error Message of FOREIGN KEY REFERENCES

# CHECK

- A CHECK constraint limits the values that can be placed in a column.
  - Check constraints enforce integrity of data.
- 
- A CHECK constraint operates by specifying a search condition, which can evaluate to TRUE, FALSE, or unknown.
  - Values that evaluate to FALSE are rejected.



The screenshot shows a 'Messages' window with the following text:

```
Msg 547, Level 16, State 0, Line 1
The INSERT statement conflicted with the CHECK constraint
"CK_NewEMP_Ph_Amount_48EFCE0F". The conflict occurred in
database "AdventureWorks2019",
table "dbo.NewEMP_PhoneExpenses", column 'Amount'.
The statement has been terminated.
```

**Output Error Message of CHECK Constraint**



# NOT NULL

- A NOT NULL constraint enforces that the column will not accept null values.
- The NOT NULL constraints are used to enforce domain integrity, similar to CHECK constraints.

# Summary

- Most tables have a primary key, made up of one or more columns of the table that identifies records uniquely.
- The nullability feature of a column determines whether rows in the table can contain a null value for that column.
- A DEFAULT definition for a column can be created at the time of table creation or added at a later stage to an existing table.
- The IDENTITY property of SQL Server is used to create identifier columns that can contain auto-generated sequential values to uniquely identify each row within a table.
- Constraints are used to apply business logic rules and enforce data integrity.
- A UNIQUE constraint is used to ensure that only unique values are entered in a column or set of columns.
- A foreign key in a table is a column that points to a primary key column in another table.
- A CHECK constraint limits the values that can be placed in a column.