



Alter and Drop Statements



ALTER Command

- ALTER command is used to change the structure of a database object
- Alter statements
 - ALTER DATABASE
 - ALTER TABLE
 - ALTER VIEW
 - ALTER PROC
 - Etc.



ALTER TABLE

- The ALTER TABLE statement is used to
 - add, delete, or modify columns in an existing table.
 - add and drop various constraints on an existing table
- This includes:
 - Adding a new column
 - Dropping an existing column
 - Changing a data type of a column
 - Adding a new constraint
 - Dropping an existing constraint
 - Enabling / Disabling a constraint
- Any change that we make to a table should first agree with the data that is already stored in it



ALTER TABLE (syntax)

ALTER TABLE table_name

{

[ALTER COLUMN <column_name> <new_data_type>
[NULL|NOT NULL]

| ADD <column name> <data_type> ...

| ADD <column_name> AS <computed_column_exp>

| DROP COLUMN <column_name>

| ADD CONSTRAINT <constraint_name>

| DROP CONSTRAINT <column_name>

}



ALTER COLUMN

- Syntax

ALTER TABLE *TableName*

ALTER COLUMN *columnName ColDef*

- Alter Column allows you to perform some changes to an existing column of a table
- Using Alter Column, you can:
 - ☐ Modify the data type
 - ☐ Change the size
 - ☐ Add a NOT NULL constraint



ALTER COLUMN – (cont.)

- When using ALTER Column, the modified column cannot be:
 - ☐ A computed column or used in a computed column.
 - ☐ Used in an index
 - ☐ Used in a PRIMARY KEY or [FOREIGN KEY] REFERENCES constraint
 - ☐ Used in a CHECK or UNIQUE constraint
 - ☐ Associated with a default definition
- In general, other columns or constraints should not be dependent on the column that is being modified



ALTER COLUMN – (cont.)

■ Changing Nullability

- NULL can be specified in ALTER COLUMN to force a NOT NULL column to allow null values
- Nullability cannot be changed to NULL for columns in PRIMARY KEY constraints
- NOT NULL can be specified in ALTER COLUMN only if the column contains no null values.

■ Changing Nullability requires that you specify the data type, even if it is not changed

■ Example

ALTER TABLE *Employee*

ALTER COLUMN *LastName* VARCHAR (20) NOT NULL



ALTER COLUMN – Examples

- Changing the size of a column

```
ALTER TABLE Student  
ALTER COLUMN FName varchar(25)
```

- Changing column collation

```
ALTER TABLE Student  
ALTER COLUMN FName nvarchar(50) COLLATE Latin1_General_100_CS_AI
```

- Changing the data type of a column

```
ALTER TABLE Employee  
ALTER COLUMN Salary DECIMAL (5, 2)
```



ADD Column

- Syntax

ALTER TABLE *TableName*
ADD *ColDefn*

- Columns that do not allow NULL values can be added with ALTER TABLE only if
 - ☐ they have a default specified or
 - ☐ the table is empty
- Multiple columns can be added in a single statement
- Columns can be added along with their constraint
- What IF
 - ☐ What if the table has some data?
 - ☐ What if the column allows NULL and has a DEFAULT ?



ADD Column - Examples

- Adding a column to an existing table

```
ALTER TABLE Employee  
ADD          GrandFatherName varchar(50) NULL
```

- Adding multiple columns at one time

```
ALTER TABLE Employee  
ADD      GrandFatherName varchar(50) NULL  
        , DateOfEmployment date NOT NULL DEFAULT GetDate()
```

- What if the *Employee* table has some data?

- ☐ All existing rows of the *Employee* table will have
 - NULL for the *GrandFatherName* column
 - Will have the CURRENT DATE for the *DateOfEmployment* column



Nulls and Defaults

- If a new column **allows NULL** and **no DEFAULT** is specified
 - the new column contains a null value for each row in the table
- If the new column **allows NULL** values and a **DEFAULT** definition is added with the new column
 - **WITH VALUES** must be used to store the default value in the new column for each existing row in the table
- If the new column does not allow null values and the table is not empty
 - a **DEFAULT** definition must be added with the new column, and
 - the new column automatically loads with the default value in the new columns in each existing row.



ADD CONSTRAINT

- Syntax

```
ALTER TABLE TableName  
ADD Constraint ConstraintDefn
```

- ADD CONSTRAINT allows you to add a new constraint to an existing column
- Existing data should agree with the new constraint

- Example

```
ALTER TABLE Customers  
ADD CONSTRAINT DefaultGfname DEFAULT 'UNKNOWN' FOR Gfname
```



ALTER TABLE and Views

Question:

- What happens to your views when you add a new column to a table?
 - ☐ The NEW columns will not be accessible via the view
 - ☐ Even if your view is built using a SELECT * as its base statement
- Column names in views are resolved at the time the view is created
- You must either DROP and recreate the view or
 - ☐ use an ALTER VIEW statement to rebuild it



DROP Statement

- A DROP is the same as deleting or removing an object from the database
- The syntax is the same for all of the database objects
 - Tables, views, stored procedures, triggers, and so on

DROP <object type> <object name> [, ...n]

DROP DATABASE *databaseName*

DROP TABLE *tableName*



DROP Statement (cont.)

- You use the DROP statement in SQL to delete created database objects
- You can use the DROP statement to delete databases, tables and indexes
- DROP statement can also be used in conjunction with the ALTER TABLE statement to delete fields from tables



DROP TABLE

- DROP TABLE removes one or more tables
- All data, indexes, constraints, and other objects defined on the table will be deleted
- Any view or stored procedure that references the dropped table must be explicitly dropped
- DROP TABLE cannot be used to drop a table if the table is referenced by a FOREIGN KEY constraint
- Examples

DROP TABLE ***Customers, Employees***

DROP TABLE IF EXISTS ***sales***



Examples 1

- Adding a new column

```
ALTER TABLE tblTest  
ADD columnB VARCHAR(20) NULL
```

In the new column, each row will have NULL

- Adding a column with a constraint

```
ALTER TABLE tblTest  
ADD colB VARCHAR(20) NULL CONSTRAINT colBUnique UNIQUE
```

- Adding a DEFAULT constraint to an existing column

```
ALTER TABLE tblTest  
ADD CONSTRAINT colBDefault DEFAULT 50 FOR columnB
```



Examples 2

- Adding a nullable column with default values

```
ALTER TABLE      tblTest  
ADD AddDate smalldatetime NULL  
CONSTRAINT AddDateDflt DEFAULT GETDATE() WITH VALUES
```

- WITH VALUES is used to provide values for each existing row in the table.
- If WITH VALUES is not used, each existing row will have the value NULL for the new column



Examples 3

- Dropping a column

```
ALTER TABLE tblTest  
DROP COLUMN column_b
```

- Remove multiple columns

```
ALTER TABLE tblTest  
DROP COLUMN column_c, column_d
```

- Dropping constraints

```
ALTER TABLE tblTest  
DROP my_constraint
```

OR

```
ALTER TABLE tblTest  
DROP CONSTRAINT my_constraint
```



Examples 4

- Adding a FOREIGN KEY constraint

```
ALTER TABLE tblTest  
ADD CONSTRAINT FK_ContactBackup_Contact  
FOREIGN KEY (ContactID) REFERENCES Person (BusinessEntityID)
```

- Dropping a FOREIGN KEY constraint

```
ALTER TABLE tblTest  
DROP CONSTRAINT FK_ContactBackup_Contact
```

- Adding an unverified CHECK constraint to an existing column

```
ALTER TABLE tblTest  
WITH NOCHECK  
ADD CONSTRAINT exdCheck CHECK (columnA > 1)
```