https://github.com/ManjDesp/sqlscripts.git

---

* TO CREATE DATABASE

```
SYNTAX:         CREATE DATABASE DATABASE_NAME;

EX:                     CREATE DATABASE testDB;
```

---

* TO DROP DATABASE

```
SYNTAX:         DROP DATABASE DATABASE_NAME;

EG:                     DROP DATABASE testDB;
```

---

* BACKUP DATABASE: Statement is used in SQL Server to create a full back up of an existing SQL database.

```
SYNTAX:         BACKUP DATABASE databasename TO DISK = 'filepath';

EG :1:          BACKUP DATABASE testDB TO DISK = 'C:\MyDatabase';

ERROR:          Msg 3201, Level 16, State 1, Line 1
                        Cannot open backup device 'C:\MyDatabase'. Operating system error 5(Access is
denied.).
                        Msg 3013, Level 16, State 1, Line 1
                        BACKUP DATABASE is terminating abnormally.

                        Completion time: 2022-10-05T08:29:19.1712701+05:30

EG :2:          BACKUP DATABASE testDB TO DISK = 'D:\RECYCLE BIN EXTRAS\CSHARP';

ERROR:          Msg 3201, Level 16, State 1, Line 1
                        Cannot open backup device 'D:\RECYCLE BIN EXTRAS\CSHARP'. Operating system error
5(Access is denied.).
                        Msg 3013, Level 16, State 1, Line 1
                        BACKUP DATABASE is terminating abnormally.
```

```
                        Completion time: 2022-10-05T08:32:54.4461566+05:30

-------------------------------------------------------------------------------
* BACKUP WITH DIFFERENTIAL:
-------------------------------------------------------------------------------

SYNTAX:         BACKUP DATABASE databasename TO DISK = 'filepath'
                        WITH DIFFERENTIAL;

EX:                     BACKUP DATABASE testDB TO DISK = 'D:\RECYCLE BIN
EXTRAS\CSHARP\MyDatabase\mydata.bak'
                        WITH DIFFERENTIAL;

ERROR:          Msg 3035, Level 16, State 1, Line 1
                        Cannot perform a differential backup for database "testDB",
                        because a current database backup does not exist.
                        Perform a full database backup by reissuing BACKUP DATABASE, omitting the WITH
DIFFERENTIAL option.
                        Msg 3013, Level 16, State 1, Line 1
                        BACKUP DATABASE is terminating abnormally.

                        Completion time: 2022-10-05T08:38:24.0044830+05:30

POINTS:   * A differential back up reduces the back up time (since only the changes are backed up)

-------------------------------------------------------------------------------
* TO CREATE TABLE:
-------------------------------------------------------------------------------

SYNTAX:         CREATE TABLE table_name (
                                column1 datatype,
                                column2 datatype,
                                column3 datatype,
                                ....
                        );

COLUMN PARAMETERS:
```

Specify the names of the columns of the table.

DATATYPE PARAMETERS:

Specifies the type of data the column can hold.(e.g. varchar, integer, date, etc.)

```
EX:                     CREATE TABLE testdata(
                                testid   int,
                                testna   varchar(50),
                                tesla    varchar(50),
                                address  varchar(200),
                                city     varchar(30)
                        );
```

---

* TO CREATE TABLE USING ANOTHER TABLE

```
SYNTAX:         CREATE TABLE new_table_name AS
                        SELECT column1, column2,...
                        FROM existing_table_name
                        WHERE ....;

EX:                     CREATE TABLE dbtestdata AS
                        SELECT testna,city
                        FROM testdata;


ERROR 1:        Msg 156, Level 15, State 1, Line 2 Incorrect syntax near the keyword 'SELECT'.
                        Completion time: 2022-10-05T09:03:53.5574359+05:30

ERROR 2:        Incorrect Syntax near 'SELECT'.Expecting EDGE_TYPE or FILETABLE.
```

---

* TO DROP TABLE

```
SYNTAX:         DROP TABLE table_name;

EX:                     DROP TABLE testdata;
```

```
* TO TRUNCATE TABLE
-------------------------------------------------------------------------------

SYNTAX:         TRUNCATE TABLE table_name;

EX:                     TRUNCATE TABLE testdata;
-------------------------------------------------------------------------------
* TO INSERT INTO TABLE
-------------------------------------------------------------------------------

*  It is possible to write the INSERT INTO statement in two ways:

1: Specify both the column names and the values to be inserted:
-----------------------------------------------------------------
        * SYNTAX:               INSERT INTO table_name (column1, column2, column3, ...)
                                        VALUES (value1, value2, value3, ...);

        * EX 1:                 INSERT INTO testdata(testid,testna,tesla,address,city)
                                        VALUES (1,'Manoj','MJ','Thygarajnagar','Bangalore Rural');

                                        INSERT INTO testdata(testid,testna,tesla,address,city)
                                        VALUES (2,'Rahul','RJ','Shivajinagar','Bangalore Urban');

        * EX 2:                 INSERT INTO testdata(testid,testna,city)
                                        VALUES (3,'Ravikanth','Shimoga');

                                 -- THE OTHER FIELDS WILL BE NULL.


2: If you are adding values for all the columns of the table, you do not need to specify the column names in
the
   SQL query. However, make sure the order of the values is in the same order as the columns in the table.
-----------------------------------------------------------------
        * SYNTAX:               INSERT INTO table_name
                                        VALUES (value1, value2, value3, ...);

        * EX:                   INSERT INTO testdata
                                        VALUES (1,'Manoj','MJ','Thygarajnagar','Bangalore Rural');
```

```
                                        INSERT INTO testdata
                                        VALUES (2,'Rahul','RJ','Shivajinagar','Bangalore Urban');

                                        INSERT INTO testdata
                                        VALUES (3,'Ravikanth','RA','Magadi Road','Tumkur');

                                        INSERT INTO testdata
                                        VALUES (4,'Raja','RAJ','Indian Gall Express','Mumbai');

                                        INSERT INTO testdata
                                        VALUES (5,'Sahegal','SAL','North ISI','Hyderbad');
-------------------------------------------------------------------------------
* TO TRUNCATE TABLE: It is used to delete the data inside a table, but not the table itself.
-------------------------------------------------------------------------------

SYNTAX:         TRUNCATE TABLE table_name;

EX:                     TRUNCATE TABLE testdata;
-------------------------------------------------------------------------------
* TO ALTER TABLE:
-------------------------------------------------------------------------------

* It is used to add, delete, or modify columns in an existing table.

* It is also used to add and drop various constraints on an existing table.


1: ALTER TABLE - ADD Column
-----------------------------------------------------------------
        * SYNTAX:               ALTER TABLE table_name ADD column_name datatype;

        * EX    :               ALTER TABLE testdata ADD temail varchar(100);

2: ALTER TABLE - DROP Column
-----------------------------------------------------------------
        * SYNTAX:               ALTER TABLE table_name
                                        DROP COLUMN column_name;

        * EX    :               ALTER TABLE testdata
```

```
                                        DROP COLUMN address;

3: ALTER TABLE - ALTER/MODIFY COLUMN : To change the data type of a column in a table :(THIS IS FOR MS-SQL)
-------------------------------------------------------------------
        * SYNTAX:               ALTER TABLE table_name
                                        ALTER COLUMN column_name datatype;

        * EX    :               ALTER TABLE testdata ADD tdate varchar(100);

                                        ALTER TABLE testdata
                                        ALTER COLUMN tdate date;
-------------------------------------------------------------------------------------
* TO CREATE CONSTRAINTS:
-------------------------------------------------------------------------------------

* SQL constraints are used to specify rules for the data in a table.

* Used to limit the type of data that can go into a table.This ensures the accuracy and reliability of the
  data in the table.

* If there is any violation between the constraint and the data action::The action is aborted.

* It can be column level or table level.

* Column level constraints: Apply to a column :: Table level constraints : Apply to the whole table.

* CONSTRAINTS:
---------------
1:NOT NULL:             Ensures that a column cannot have a NULL value

2:UNIQUE  :             Ensures that all values in a column are different.

3:PRIMARY KEY:  A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table.

4:FOREIGN KEY:  Prevents actions that would destroy links between tables.

5:CHECK:                Ensures that the values in a column satisfies a specific condition.

6:DEFAULT:              Sets a default value for a column if no value is specified.
```

7:CREATE INDEX: Used to create and retrieve data from the database very quickly.

---

* NOT NULL

---

* This enforces a field to always contain a value, which means that you cannot insert a new record,
  or update a record without adding a value to this field.

```
* EG:                   CREATE TABLE testdata1(
                                testid   int                    NOT NULL,
                                testna   varchar(50)    NOT NULL,
                                tesla    varchar(50)    NOT NULL,
                                address  varchar(200)   NOT NULL,
                                city     varchar(30)
                        );
```

---

* NOT NULL ON ALTER TABLE - MS SQL

```
* SYNTAX:               ALTER TABLE table_name
                                ALTER COLUMN column_name datatype constraint;

* EG:                   ALTER TABLE testdata1
                                ALTER COLUMN city varchar(30) NOT NULL;
```

---

* UNIQUE - MS SQL

---

* The UNIQUE constraint ensures that all values in a column are different.

* Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.

* A PRIMARY KEY constraint automatically has a UNIQUE constraint.

* However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

```
* EG:                  CREATE TABLE testdata1(
                               testid   int                    NOT NULL        UNIQUE,
                               testna   varchar(50)    NOT NULL    UNIQUE,
                               tesla    varchar(50),
                               address  varchar(200),
                               city     varchar(30)
                       );
--------------------------
* CASE1:               INSERT INTO testdata1
                               VALUES (1,'Manoj','MJ','Thygarajnagar','Bangalore Rural');

                               INSERT INTO testdata1
                               VALUES (1,'Rohan','RJ','Shivajinagar','Bangalore Urban');

* ERROR:               (1 row affected)
                               Msg 2627, Level 14, State 1, Line 4
                               Violation of UNIQUE KEY constraint 'UQ__testdata__A29AFFE1FF796156'.
                               Cannot insert duplicate key in object 'dbo.testdata1'. The duplicate key
value is (1).
                               The statement has been terminated.

                               Completion time: 2022-10-05T11:36:24.9452812+05:30
--------------------------
* CASE2:               INSERT INTO testdata1
                               VALUES (2,'Manoj','MJ','Thygarajnagar','Bangalore Rural');

                               INSERT INTO testdata1
                               VALUES (3,'Manoj','RJ','Shivajinagar','Bangalore Urban');

* ERROR1:              Msg 2627, Level 14, State 1, Line 1
                               Violation of UNIQUE KEY constraint 'UQ__testdata__A29B42544C80F823'.
                               Cannot insert duplicate key in object 'dbo.testdata1'. The duplicate key
value is (Manoj).
                               The statement has been terminated.
                               Msg 2627, Level 14, State 1, Line 4
                               Violation of UNIQUE KEY constraint 'UQ__testdata__A29B42544C80F823'.
                               Cannot insert duplicate key in object 'dbo.testdata1'. The duplicate key
value is (Manoj).
```

```
                                The statement has been terminated.

                                Completion time: 2022-10-05T11:43:38.0840025+05:30
------------------------
* CASE3:                INSERT INTO testdata1
                                VALUES (2,'Manya','MJ','Thygarajnagar','Bangalore Rural');

                                INSERT INTO testdata1
                                VALUES (3,'Manya','RJ','Shivajinagar','Bangalore Urban');

* ERROR2:               (1 row affected)
                                Msg 2627, Level 14, State 1, Line 4
                                Violation of UNIQUE KEY constraint 'UQ__testdata__A29B42544C80F823'.
                                Cannot insert duplicate key in object 'dbo.testdata1'. The duplicate key
value is (Manya).
                                The statement has been terminated.

                                Completion time: 2022-10-05T11:47:08.8799464+05:30

-------------------------------------------------------------------------------
* UNIQUE - To name a UNIQUE constraint, and to define a UNIQUE constraint on multiple columns,
-------------------------------------------------------------------------------

* SYNTAX:               CONSTRAINT constraint_name UNIQUE (col1,col2,col3....coln)

                                Here constraint_name is not type, but to name UNIQUE constraint.

* EG:                   CREATE TABLE testdata1(
                                testid          int                             NOT NULL,
                                testna          varchar(50)             NOT NULL,
                                tesla           varchar(50),
                                address         varchar(200),
                                age                     int,
                                CONSTRAINT  idna  UNIQUE (testid,testna)
                        );
-------------------------------------------------------------------------------
* UNIQUE CONSTRAINT ON ALTER TABLE
-------------------------------------------------------------------------------
```

* SYNTAX: ALTER TABLE table_name ADD UNIQUE (colname);

* EG: ALTER TABLE testdata1 ADD UNIQUE(tesla);

---

* UNIQUE CONSTRAINT - To name a UNIQUE constraint, and to define a UNIQUE constraint on multiple column

---

* SYNTAX: ALTER TABLE table_name ADD CONSTRAINT constraint_name UNIQUE (col1,col2,..coln);

* EG: ALTER TABLE testdata1   ADD CONSTRAINT myc1 UNIQUE(testid,testna);

Here constraint_name is not type, but to name UNIQUE constraint.

---

* DROP UNIQUE CONSTRAINT

---

* SYNTAX: ALTER TABLE table_name
Here constraint_name is not type, but to name UNIQUE constraint.

---

* PRIMARY KEY

---

* The PRIMARY KEY constraint uniquely identifies each record in a table.

* Primary keys must contain UNIQUE values, and cannot contain NULL values.

* A table can have only ONE primary key and in the table, this primary key can consist of single or multiple columns (fields).

---

* PRIMARY KEY ON CREATE TABLE

```
--------------------------------------------------------------------------------

* EG:                   CREATE TABLE testdata1(
                                testid          int                             NOT NULL        PRIMARY KEY,
                                testna          varchar(50)             NOT NULL,
                                tesla           varchar(50),
                                address         varchar(200),
                                age                     int,
                        );
--------------------------------------------------------------------------------
* PRIMARY KEY - To allow naming of a PRIMARY KEY constraint, and for defining a PRIMARY KEY constraint
                                on multiple columns.
--------------------------------------------------------------------------------

* SYNTAX:               CONSTRAINT constraint_name PRIMARY KEY (col1,col2,col3....coln)

                                Here constraint_name is not type, but to name PRIMARY KEY constraint.

* EG:                   CREATE TABLE testdata1(
                                testid          int                             NOT NULL,
                                testna          varchar(50)             NOT NULL,
                                tesla           varchar(50),
                                address         varchar(200),
                                age                     int,
                                CONSTRAINT  pk_test  PRIMARY KEY (testid,testna)
                        );

*NOTE:                  In the example above there is only ONE PRIMARY KEY (pk_test).

                                However, the VALUE of the primary key is made up of TWO COLUMNS (testid +
testna).

--------------------------------------------------------------------------------
* PRIMARY KEY CONSTRAINT ON ALTER TABLE
--------------------------------------------------------------------------------

* REFERENCE:    CREATE TABLE testdata1(
                                testid          int                             NOT NULL,
                                testna          varchar(50)             NOT NULL,
```

```
                        tesla           varchar(50),
                        address         varchar(200),
                        age                     int,
                        somali          varchar(20),
                        mya                     varchar(30),
                        datadat         date,
                        mysla           int
                );
```

```
* SYNTAX:               ALTER TABLE table_name  ADD PRIMARY KEY (colname);

* EG:                   ALTER TABLE testdata1 ADD PRIMARY KEY(testid);
```

```
* PRIMARY KEY CONSTRAINT -  To name a PRIMARY KEY constraint, and to define a PRIMARY KEY constraint on
                                                multiple column
```

```
* SYNTAX:               ALTER TABLE table_name  ADD CONSTRAINT  constraint_name PRIMARY KEY
(col1,col2,..coln);

* EG:                   ALTER TABLE testdata1   ADD CONSTRAINT  mypk_nage PRIMARY KEY (testid,testna);

                        Here constraint_name is not type, but to name PRIMARY KEY constraint.

* NOTE:                 If you use ALTER TABLE to add a primary key, the primary key column(s) must have
been declared to
                        not contain NULL values (when the table was first created).
```

```
* DROP PRIMARY KEY CONSTRAINT
```

```
* SYNTAX:               ALTER TABLE table_name
                        DROP CONSTRAINT constraint_name;

* EG:                   ALTER TABLE testdata1
```

```
                                DROP CONSTRAINT mypk_nage;

                                Here constraint_name is not type, but to name PRIMARY KEY constraint.
--------------------------------------------------------------------------------
* FOREIGN KEY
--------------------------------------------------------------------------------

* FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

* A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another

  table.

* The table with the foreign key is called the Child table.

* The table with the primary key is called the Referenced or Parent table.

--------------------------------------------------------------------------------
* FOREIGN KEY ON CREATE TABLE
--------------------------------------------------------------------------------
* REFERENCE:
 -----------
* TABLE1:                               CREATE TABLE tdutta(
                                                       dutid           int                             NOT
NULL,
                                                       dutna           varchar(50)             NOT NULL,
                                                       dutdd           varchar(50)             NOT NULL,
                                                       testid          int
FOREIGN KEY     REFERENCES      testdata1(testid)               <----
                                                    );

* TABLE2:                               CREATE TABLE testdata1(
                                                       testid          int                             NOT
NULL    PRIMARY KEY,
                                                       testna          varchar(50)             NOT NULL,
                                                       tesla           varchar(50),
                                                       address         varchar(200),
                                                       age                     int,
                                                       somali          varchar(20),
```

```
                                                    mya                     varchar(30),
                                                    datadat         date,
                                                    mysla           int
                                                );
--------------------------------------------------------------------------------
* FOREIGN KEY - To allow naming of a FOREIGN KEY constraint, and for defining a FOREIGN KEY constraint
                                on multiple columns.
--------------------------------------------------------------------------------
* CONCEPTS:
--------------------------------------------------------------------------------
* SYNTAX:                CREATE TABLE childTable
                                (
                                  column_1 datatype [ NULL |NOT NULL ],
                                  column_2 datatype [ NULL |NOT NULL ],
                                  ...

                                 CONSTRAINT constraint_name
                                        FOREIGN KEY (child_column1, child_column2, ... child_column_n)
                                        REFERENCES parentTable (parent_column1, parent_column2, ...
parent_column_n)
                                        [ ON DELETE { NO ACTION |CASCADE |SET NULL |SET DEFAULT } ]
                                        [ ON UPDATE { NO ACTION |CASCADE |SET NULL |SET DEFAULT } ]
                                );

                                Here constraint_name is not type, but to name FOREIGN KEY constraint.

* ON DELETE:    An optional parameter. It specifies what happens to the child data after deletion of the
parent
                                data.

                                Some of the values for this parameter include

                                *NO ACTION
                                *SET NULL
                                *CASCADE,
                                *SET DEFAULT.

*ON UPDATE:             An optional parameter. It specifies what happens to the child data after update on
the parent
```

data.

Some of the values for this parameter include

*NO ACTION
*SET NULL
*CASCADE
*SET DEFAULT.

*NO ACTION: used together with ON DELETE and ON UPDATE.

It means that nothing will happen to the child data after the update or deletion of the parent
data.

*CASCADE: used together with ON DELETE and ON UPDATE.
The child data will either be deleted or updated after the parent data has been deleted or updated.

*SET NULL: used together with ON DELETE and ON UPDATE.
The child will be set to null after the parent data has been updated or deleted.

*SET DEFAULT: used together with ON DELETE and ON UPDATE.
The child data will be set to default values after an update or delete on the parent data.

---

* FOREIGN KEY - To allow naming of a FOREIGN KEY constraint, and for defining a FOREIGN KEY constraint on multiple columns.

---

* STEPS:

* Parent Table: Say, we have an existing Parent table as ‘COURSE.’
Course_ID and Course_name are two columns with Course_Id as Primary Key.

* Child Table : We need to create the second table as a child table.
‘Course_ID’ and ‘Course_Strength’ as two columns. However, ‘Course_ID’ shall

be Foreign Key.

* Parent Table:

```
CREATE TABLE COURSE
(
        Course_Id               INT                             NOT NULL
PRIMARY KEY,
        Course_Name             VARCHAR(20)
);
```

* Child Table:

```
CREATE TABLE COURSE_STRENGTH
(
        Course_Id                       INT                     NOT NULL,
        Course_Strength         VARCHAR(50)
);
```

* STEPS:

* Right Click on Tables>New> Table….

* Enter two column name as ‘Course_ID’ and ‘Course_Strength.’

* Right click on ‘Course_Id’ Column. Now click on Relationship.

* In ‘Foreign Key Relationship,’ Click ‘Add’.

* In ‘Table and Column Spec’ click on ‘…’ icon.

* Select ‘Primary Key Table’ as ‘COURSE’ and
  the new table now being created as ‘Foreign Key Table’ from the drop down.

* Primary Key Table’ – Select ‘Course_Id’ column as ‘Primary Key table’
column.

 ‘Foreign Key Table’ - Select ‘Course_Id’ column as ‘Foreign Key table’
column.

* Click OK.

* Click on Add.

* Click on Close.

* Give the Table name as ‘Course_Strength’ and click on OK.

* If the Pop Up box comes and say do you want to save data,then Click Yes/Ok.

---

* FOREIGN KEY ON ALTER TABLE

---

* SYNTAX: ALTER TABLE table_name ADD FOREIGN KEY (colname) REFERENCES table_name(colname);

* EG : ALTER TABLE testdata1 ADD FOREIGN KEY (testid) REFERENCES testdata(testid);

---

* DROP FOREIGN KEY

---

* SYNTAX: ALTER TABLE table_name
DROP CONSTRAINT constraint_name;

* EG: ALTER TABLE testdata1
DROP CONSTRAINT myfk_nage;

Here constraint_name is not type, but to name PRIMARY KEY constraint.

---

* CHECK CONSTRAINT

---

* The CHECK constraint is used to limit the value range that can be placed in a column.

* If you define a CHECK constraint on a column it will allow only certain values for this column.

* If you define a CHECK constraint on a table it can limit the values in certain columns based

```
  on values in other columns in the row.

--------------------------------------------------------------------------------
* CHECK ON CREATE TABLE
--------------------------------------------------------------------------------

* EG:                           CREATE TABLE testcheck(
                                        checkid         int                             NOT NULL
PRIMARY KEY,
                                        checkna         varchar(50)             NOT NULL,
                                        checkla         varchar(50),
                                        checess         varchar(200),
                                        checage         int                             CHECK
(checage>=20)
                                    );

*TRY INSERTING          INSERT INTO testcheck
 VALUES:                        VALUES (1,'Manoj','MJ','Thygarajnagar',20);

                                    INSERT INTO testcheck
                                    VALUES (2,'Rahul','RJ','Shivajinagar', 25);

                                    INSERT INTO testcheck
                                    VALUES (3,'Sagar','SAG','Chickpet', 12);
<---------THIS WONT WORK;ERROR


*ERROR:                         Msg 547, Level 16, State 0, Line 1
                                    The INSERT statement conflicted with the CHECK constraint
"CK__testcheck__checa__160F4887".
                                    The conflict occurred in database "testDB", table "dbo.testcheck",
column 'checage'.
                                    The statement has been terminated.

                                    Completion time: 2022-10-05T18:38:15.7852965+05:30

--------------------------------------------------------------------------------
* CHECK ON CREATE TABLE: To allow naming of a CHECK constraint, and for defining a CHECK constraint on
```

multiple columns.

---

* SYNTAX: CONSTRAINT constraint_name CHECK (cond1,cond2....)

* EG:

```
CREATE TABLE testcheck(
        checkid         int                             NOT NULL
PRIMARY KEY,
        checkna         varchar(50)             NOT NULL,
        checkla         varchar(50),
        checess         varchar(200),
        checage         int
        CONSTRAINT chk_test CHECK (checage>=5 AND checkla='MJ')
    );
```

Here constraint_name is not type, but to name CHECK constraint.

---

* CHECK ON ALTER TABLE

---

* SYNTAX: ALTER TABLE table_name ADD CHECK (condition);

* EG : ALTER TABLE testdata1 ADD CHECK (checage>=5);

---

* CHECK ON ALTER TABLE: To allow naming of a CHECK constraint, and for defining a CHECK constraint on multiple columns

---

* SYNTAX: ALTER TABLE table_name ADD CONSTRAINT constraint_name (cond1,cond2....);

* EG : ALTER TABLE testdata1 ADD CONSTRAINT chk_test(checage>=5 AND checkla='RJ');

Here constraint_name is not type, but to name CHECK constraint.

```
* DROP CHECK CONSTRAINT
-------------------------------------------------------------------------------

* SYNTAX:               ALTER TABLE table_name
                                DROP CONSTRAINT constraint_name;

* EG:                   ALTER TABLE testdata1
                                DROP CONSTRAINT chk_test;

                                Here constraint_name is not type, but to name CHECK constraint.

-------------------------------------------------------------------------------
* DEFAULT CONSTRAINT
-------------------------------------------------------------------------------

* The DEFAULT constraint is used to set a default value for a column.

* The default value will be added to all new records, if no other value is specified.

* It can also be used to insert system values, by using functions like GETDATE():

-------------------------------------------------------------------------------
* DEFAULT ON CREATE TABLE
-------------------------------------------------------------------------------

* EG:                           CREATE TABLE testdef(
                                        checkid         int                             NOT NULL
PRIMARY KEY,
                                        checkna         varchar(50)             NOT NULL,
                                        checkla         varchar(50)     DEFAULT         'NO',
                                        checess         varchar(200),
                                        checage         int
                                );

TRY INSERTING           INSERT INTO testdef
VALUES:                         VALUES (1,'Manoj','','Thygarajnagar',20);   -- EMPTY FIELD

                                        INSERT INTO testdef
                                        VALUES (2,'ManojP','Thygarajnagar',20);
```

```
                                        ERROR:  Msg 213, Level 16, State 1, Line 1
                                                Column name or number of supplied values does not
match table definition.

                                                Completion time: 2022-10-05T19:08:18.7692290+05:30

                                        INSERT INTO testdef(checkid,checkna,checess,checage)
<--------------TRY THIS SOLUTION
                                        VALUES (3,'ManojR','ThySim',25);

-------------------------------------------------------------------------------
* DEFAULT ON ALTER TABLE
-------------------------------------------------------------------------------

* SYNTAX:                       ALTER TABLE table_name ADD CONSTRAINT constraint_name
                                        DEFAULT 'value' FOR column_name;

* EG    :                       ALTER TABLE testdef ADD CONSTRAINT check_lal
                                        DEFAULT 18      FOR checage;

-------------------------------------------------------------------------------
* DROP DEFAULT CONSTRAINT
-------------------------------------------------------------------------------

* SYNTAX:                       ALTER TABLE table_name
                                        DROP  CONSTRAINT constraint_name;

* EG:                           ALTER TABLE testdef
                                        DROP  CONSTRAINT check_lal;
-------------------------------------------------------------------------------
* CREATE INDEX CONSTRAINT
-------------------------------------------------------------------------------

* It  is used to create indexes in tables.

* Indexes are used to retrieve data from the database more quickly than otherwise.

  The users cannot see the indexes, they are just used to speed up searches/queries.
```

---

* CREATE INDEX:

---

* Creates an index on a table. Duplicate values are allowed

* SYNTAX: CREATE INDEX index_name
ON table_name (column1, column2, ...);

* EG: CREATE INDEX person_index_one
ON Persons(LastName,FirstName);

---

* CREATE UNIQUE INDEX:

---

* Creates a unique index on a table. Duplicate values are not allowed.

* The syntax for creating indexes varies among different databases. Therefore Check the syntax for creating

indexes in your database.

* SYNTAX: CREATE UNIQUE INDEX index_name
ON table_name (column1, column2, ...);

* EG: CREATE UNIQUE INDEX person_index_two
ON Persons(Age);

---

* DROP INDEX:

---

* It is used to delete an index in a table.

* SYNTAX: DROP INDEX table_name.index_name;

* EG: DROP INDEX Persons.person_index_two;

---

* AUTO INCREMENT:

---

* Auto-increment allows a unique number to be generated automatically when a new record is inserted into a table.

* Often this is the primary key field that we would like to be created automatically every time a new record is

  inserted.

* EG:

```
CREATE TABLE Persons (
    Personid        int IDENTITY(1,1) PRIMARY KEY,
    LastName        varchar(255) NOT NULL,
    FirstName       varchar(255),
    Age int
);
```

* The MS SQL Server uses the IDENTITY keyword to perform an auto-increment feature.

* In the example above, the starting value for IDENTITY is 1, and it will increment by 1 for each new record.

* Tip: To specify that the "Personid" column should start at value 10 and increment by 5, change it to

  IDENTITY(10,5).

* To insert a new record into the "Persons" table, we will NOT have to specify a value for the "Personid" column

  (a unique value will be added automatically):

* INSERT INTO Persons values('MJ','Bahudar',22);