USE Gomycode;

--CREATING A TABLE

CREATE TABLE Employees(

EmployeeID INT PRIMARY KEY,

Firstname VARCHAR(100),

Lastname VARCHAR(100),

Department VARCHAR(100),

Salary DECIMAL(10,2)

);

DROP TABLE EMPLOYEES;

Select \* from Employees;

--Inserting data into the table

INSERT INTO Employees(EmployeeID,Firstname,Lastname,Department,salary)

Values(1,'John','Doe','HR',55000.00),

(2,'Jane','Smith','Marketing',60000.00),

(3,'Bob','Johnson','IT',65000.00);

---VIEW THE STRUCTURE OF THE TABLE

SP\_HELP EMPLOYEES;

----RETRIEVE ALL EMPLOYEE FROM THE EMPLOYEE TABLE

SELECT EmployeeID,Firstname FROM Employees;

--Retrieve employee in the Hr department

SELECT \* FROM Employees WHERE department ='HR';

--Retrieve employee with a salary greater than 60000

SELECT \* FROM Employees WHERE Salary>60000;

---updating employeeID column

UPDATE Employees

SET salary=58000.00

WHERE EmployeeID=1;

--DELETE EMPLOYEE WITH ID 4

DELETE FROM Employees

WHERE EmployeeID=4;

SELECT \* FROM employees

--joins :works with the set theorys

USE Gomycode;

--create another table

CREATE TABLE Departments(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR (100)

);

SELECT \* FROM Departments;

INSERT INTO Departments( DepartmentID,DepartmentName)

VALUES(1,'HR'),

(2,'Marketing'),

(3,'IT');

--INNER JOIN

SELECT \* FROM EMPLOYEES

INNER JOIN Departments ON EMPLOYEES.EmployeeID=Departments.DepartmentID;

SELECT \* FROM Employees

INNER JOIN Departments

ON Employees.Department=departments.departmentname;

--Adding more values on the employee's table

INSERT INTO Employees(EmployeeID,Firstname,Lastname,Department,salary)

Values(4,'Andy','Fineboy','HR',75000.00),

(5,'Joseph','Smith','Marketing',80000.00),

(6,'Dopeclass','Johnson','IT',90000.00);

SELECT \* FROM Employees;

---lEFT JOIN

SELECT \* FROM Employees

LEFT JOIN Departments ON Employees.Department=Departments.Departmentname;

--Right Join

SELECT \* FROM Employees

RIGHT JOIN Departments ON Employees.Department=Departments.Departmentname;

SELECT \* FROM EMPLOYEES

LEFT JOIN Departments

ON Departments.DepartmentID=Employees.EmployeeID;

INSERT INTO Departments( DepartmentID,DepartmentName)

VALUES(7,'Finance'),

(8,'Training'),

(9,'Software');

use Gomycode;

--Aggregate Function

--Calculate the average salary of employees in It department

SELECT AVG(salary) as Average\_salary FROM employees

WHERE DEPARTMENT='IT';

---Calculate the MAXIMUM SALARY OF THE EMPLOYEES

SELECT MAX(SALARY) AS Mximum\_salary FROM EMPLOYEES;

SELECT DISTINCT(DEPARTMENT) FROM Employees;

SELECT DISTINCT(Firstname)

From Employees;

SELECT TOP 2 \* FROM

Employees;

---counting records in the table

SELECT FirstName,LastName,COUNT(\*) IT\_EMPLOYEE

FROM Employees

WHERE Department='IT'

GROUP BY FirstName,LastName;

---Subqueries

SELECT AVG(salary) AVERAGE\_SALARY FROM EMPLOYEES

SELECT EmployeeID,FirstName,LastName,Salary

FROM Employees

WHERE Salary > (SELECT AVG(Salary) FROM Employees);

--Constraints

CREATE TABLE customer(CustomerID INT PRIMARY KEY,

Firstname VARCHAR(50),lastname VARCHAR(50));

SELECT \* FROM customer;

--Creating a foreign Key

CREATE TABLE orders(OrderID INT PRIMARY KEY,CustomerID INT,

orderdate DATE,ProductName VARCHAR (100),FOREIGN KEY(CustomerID) REFERENCES Customer(CustomerID);

---CREATING A TABLE WITH A UNIQUE CONSTRAINT

CREATE TABLE products(

ProductID INT ,

ProductName VARCHAR(50)

);

INSERT INTO Products(ProductID,ProductName)

VALUES(1,'Electronics'),

(2,'Appliances'),

(3,'Furnitures'),

(4,'Cooking utensils'),

(5,'Toiletries')

ALTER TABLE Products

ADD price DECIMAL(10,2);

SELECT \* FROM Products

INSERT INTO products(price)

VALUES(200.00),

(300.00),

(150.00),

(500.00),

(600.00);

UPDATE products

SET price=200.00

WHERE ProductID=1;

UPDATE products

SET price=300.00

WHERE ProductID=2;

UPDATE products

SET price=150

WHERE ProductID=3;

UPDATE products

SET price=500

WHERE ProductID=4;

UPDATE products

SET price=600

WHERE ProductID=5;

DELETE FROM products

WHERE ProductID IS NULL;

SELECT \* FROM products;

DROP TABLE products;

---Creating a table with a check contraint

CREATE TABLE Students(

StudentID INT PRIMARY KEY,

Age INT,

CHECK (Age >= 18)

);

INSERT INTO students (StudentID,Age)

VALUES(001,16);

--Creating a table with a Default constraint

CREATE TABLE orders\_1(

OrderID INT PRIMARY KEY,

OrderDate DATE DEFAULT GETDATE()

);

INSERT INTO orders\_1(OrderID)

VALUES(1);

SELECT \* FROM orders\_1;

CREATE TABLE Parties(

Party\_id INT,

Party\_name VARCHAR(100),

Candidate\_name VARCHAR (100) DEFAULT 'NOT AVAILABLE',

Postion VARCHAR(100)

);

INSERT INTO Parties(Party\_id,Party\_name,Postion)

VALUES(001,'PDP','Presidency');

--Adding a primary key constraint to an existing table

ALTER TABLE Parties

ADD CONSTRAINT PK\_Party\_id PRIMARY KEY(party\_id);

---logical Operators

--AND example

SELECT \* FROM Employees

where Salary >= 55000 AND Department='IT';

SELECT \* FROM products;

---Using the NOT operator

SELECT \* FROM products

WHERE NOT Productname='Electronics';

SELECT \* FROM products

WHERE Productname != 'Electronics'

--Using the IN operator

SELECT \* FROM products

WHERE ProductID IN (1,2);