**LAB 06**

**TASK 2:Create the following simple table “Employee\_new” using Create table with no primary key.**

**Query:**

USE northwind;

CREATE TABLE Employee\_new (

EMPLOYEE\_ID INT,

FIRST\_NAME VARCHAR(255),

LAST\_NAME VARCHAR(255),

EMAIL VARCHAR(255),

PHONE\_NUMBER VARCHAR(20),

HIRE\_DATE DATE,

JOB\_DD VARCHAR(10),

SALARY DECIMAL(10,2),

COMMISSION\_PCT DECIMAL(5,2),

MANAGER\_ID INT

);

**TASK 3: Add the constraint on the above table:**

* **Rename the table Employee\_new to Employee\_old”**
* **Add a column Dept\_id to the table Employee\_old after email**
* **Add a Primary key Sr\_NO with auto increment**
* **Change the data type of the column phone\_number to integer**
* **Remove the Primary key constraints and add “Employee\_ID” as a primary key**
* **Add unique and not null Constraint on “EMAIL”**
* **Add a foreign key constraint named fk\_dp\_id on Dept\_id column of Dept table referencing to the primary key dept\_id**
* **Drop the table of Department**

**Query:**

ALTER TABLE Employee\_new RENAME TO Employee\_old;

ALTER TABLE Employee\_old ADD COLUMN Dept\_id INT AFTER EMAIL;

ALTER TABLE Employee\_old ADD COLUMN Sr\_NO INT AUTO\_INCREMENT PRIMARY KEY;

ALTER TABLE Employee\_old MODIFY COLUMN PHONE\_NUMBER INT;

ALTER TABLE Employee\_old DROP PRIMARY KEY,

ADD PRIMARY KEY (Employee\_ID);

ALTER TABLE Employee\_old ADD CONSTRAINT unique\_email;

ALTER TABLE Employee\_old MODIFY COLUMN EMAIL VARCHAR(255) NOT NULL;

ALTER TABLE Employee\_old ADD CONSTRAINT fk\_dp\_id **FOREIGN KEY** (Dept\_id) REFERENCES Dept(dept\_id);

DROP TABLE Department;

**Task 4: Insert 5 records of the “Employee\_old” table and update the phone number of ‘1005’**

**Query:**

INSERT INTO Employee\_old (EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_DD, SALARY, COMMISSION\_PCT, MANAGER\_ID, Dept\_id)

VALUES

(100, 'Steven', 'King', 'SKING', '515.123.4567', '1987-06-17', 'AD PRES', 24000.00, 0.00, 0),

(101, 'Neena', 'Kochhar', 'NKOCKHAR', '515.123.4568', '1987-06-18', 'AD VP', 17000.00, 0.00, 100),

(102, 'Lex', 'De Haan', 'LDEHAAN', '515.123.4569', '1987-06-19', 'AD VP', 17000.00, 0.00, 100),

(103, 'Alexander', 'Hunold', 'AHUNOLD', '590.423.4567', '1987-06-20', 'IT\_PROG', 9000.00, 0.00, 102),

(104, 'Bruce', 'Ernst', 'BERNST', '590.423.4568', '1987-06-21', 'IT\_PROG', 6000.00, 0.00, 103);

UPDATE Employee\_old

SET PHONE\_NUMBER = '590.423.4569'

WHERE EMPLOYEE\_ID = 105;

**TASK 5: Manipulating Data Across Multiple Tables.**

**1. Insert a new customer into the Customers table with CustomerID "CUST1",**

**and other relevant information.**

**2. Update the OrderDetails table to set the Quantity to 10 for all products in**

**orders made by customers located in the USA.**

**3. Delete all orders made by customers located in the USA that have not yet**

**been shipped..**

**Query:**

INSERT INTO Customers (CustomerID, CustomerName, ContactName, Address, City, PostalCode, Country)

VALUES ('CUST1', 'New Customer', 'John Doe', '123 Main St', 'Anytown', '12345', 'USA');

UPDATE OrderDetails

SET Quantity = 10

WHERE OrderID IN (

SELECT OrderID

FROM Orders

WHERE CustomerID IN (

SELECT CustomerID

FROM Customers

WHERE Country = 'USA'

)

);

DELETE FROM Orders

WHERE CustomerID IN (

SELECT CustomerID

FROM Customers

WHERE Country = 'USA'

) AND ShipDate IS NULL;

**DCL**

begin;

SAVEPOINT my\_savepoint;

SELECT \* FROM Customers WHERE CustomerID = 'bbb';

UPDATE Customers SET ContactName = 'Maria Anders' WHERE CustomerID = 'bbb';

ROLLBACK TO my\_savepoint;

SELECT \* FROM Customers WHERE CustomerID = 'bbb';

UPDATE Customers SET ContactName = 'Maria Anders' WHERE CustomerID = 'bbb';

COMMIT; /// Changes permanently saved.

ROLLBACK TO my\_savepoint;

**TASK:**

**Create a SQL transaction and a save point with your roll\_no.**

**Note: After each task create save points**

-- Start the transaction

BEGIN TRANSACTION;

-- Task 1: Insert a new product

INSERT INTO products (product\_id, product\_name, price, available\_units)

VALUES (1, 'Product A', 10.0, 100);

-- Create a savepoint after Task 1

SAVEPOINT savepoint\_123456\_task1;

-- Task 2: Update the price of a product

UPDATE products

SET price = 12.0

WHERE product\_id = 1;

-- Create a savepoint after Task 2

SAVEPOINT savepoint\_123456\_task2;

-- Task 3: Delete a product

DELETE FROM products

WHERE product\_id = 1;

-- Create a savepoint after Task 3

SAVEPOINT savepoint\_123456\_task3;

-- Rollback to savepoint after Task 2 if needed

-- ROLLBACK TO SAVEPOINT savepoint\_123456\_task2;

-- Rollback to savepoint after Task 1 if needed

-- ROLLBACK TO SAVEPOINT savepoint\_123456\_task1;

-- Commit the transaction if everything is correct

COMMIT;