Assignment 1 –

Creating Table –

CREATE TABLE Studentinfo ( Roll INT, Name VARCHAR(50), Branch VARCHAR(50), Mobile VARCHAR(50), Total INT)

Inserting Data –

INSERT INTO Studentinfo VALUES

(101, 'Ayush', 'CSE', '1234567890', 78),

(102, 'Vrushal', 'ETC', '2345678901', 89),

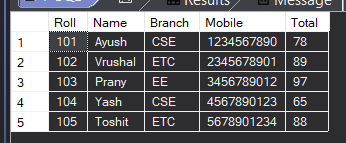
(103, 'Prany', 'EE', '3456789012', 97),

(104, 'Yash', 'CSE', '4567890123', 65),

(105, 'Toshit', 'ETC', '5678901234', 88)

Display data –

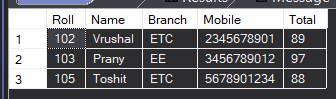
SELECT \* FROM Studentinfo;



Que 1. Display record having total > 80 using View?

CREATE VIEW total\_above\_80 AS SELECT \* FROM Studentinfo WHERE total > 80;

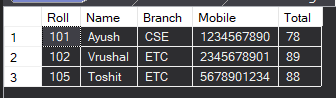
SELECT \* FROM total\_above\_80;



Que 2. Display Record where total is between 70 to 95

CREATE VIEW Bet\_total AS SELECT \* FROM Studentinfo WHERE total BETWEEN 70 AND 95

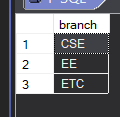
SELECT \* FROM Bet\_total



Que 3. Create view for group by branch

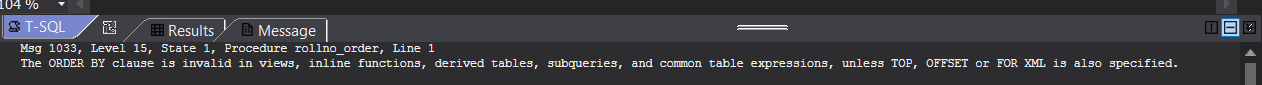
CREATE VIEW Group\_branch AS SELECT branch FROM studentinfo GROUP BY branch;

SELECT \* FROM Group\_branch;



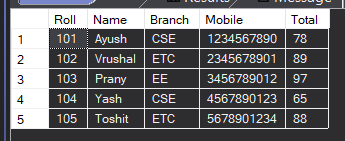
Que 4. Create view for descending Total

CREATE VIEW rollno\_order AS SELECT \* FROM studentinfo ORDER BY roll DESC;

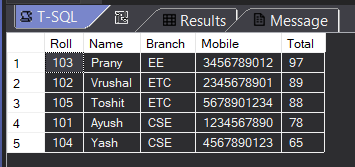


CREATE VIEW D\_studentinfo AS SELECT \* FROM studentinfo;

SELECT \* FROM D\_studentinfo;



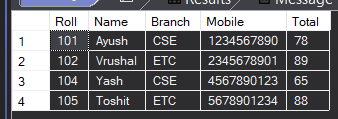
SELECT \* FROM D\_studentinfo ORDER BY total DESC;



Que 5. Create view CSE and ETC branch

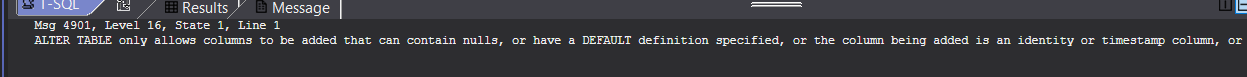
CREATE VIEW branch\_in AS SELECT \* FROM studentinfo WHERE branch IN ('CSE', 'ETC');

SELECT \* FROM branch\_in;

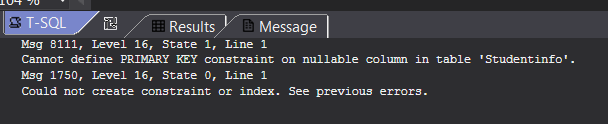


Que 6. Add new column as a primary key

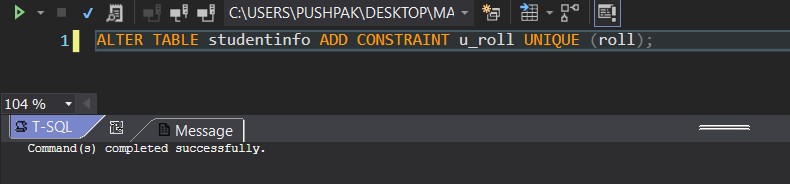
ALTER TABLE Studentinfo ADD u\_roll INT PRIMARY KEY;



ALTER TABLE Studentinfo ADD CONSTRAINT u\_roll PRIMARY KEY (roll);



Que 7. Make unique key

ALTER TABLE studentinfo ADD CONSTRAINT u\_roll UNIQUE (roll); 

Assignment 3 –

Que 1. Insert data using store procedure

Insert Data –

CREATE PROCEDURE insertdata(@id1 AS INT,

@Name1 AS VARCHAR(50),

@Email1 AS VARCHAR(50),

@Mobile1 AS VARCHAR(50),

@Salary1 AS VARCHAR(50)

)

AS

BEGIN

INSERT INTO Employee (Id, Name, Email, Mobile, salary) VALUES (@id1, @Name1, @Email1, @Mobile1, @Salary1);

END;

SQL Query –

EXEC insertdata @Id1 = 101, @Name1 = 'Ayush', @Email1 = 'ayush@gmail.com', @Mobile1 = '1234567890', @Salary1 = 40000

EXEC insertdata @Id1 = 102, @Name1 = 'Vrushal', @Email1 = 'vrushal@gmail.com', @Mobile1 = '2345678901', @Salary1 = 50000

EXEC insertdata @Id1 = 103, @Name1 = 'Yash', @Email1 = 'vrushal@gmail.com', @Mobile1 = '3456789012', @Salary1 = 20000

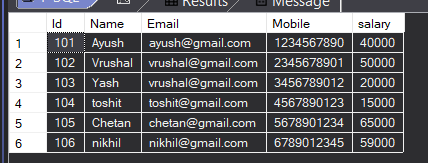
EXEC insertdata @Id1 = 104, @Name1 = 'toshit', @Email1 = 'toshit@gmail.com', @Mobile1 = '4567890123', @Salary1 = 15000

EXEC insertdata @Id1 = 105, @Name1 = 'Chetan', @Email1 = 'chetan@gmail.com', @Mobile1 = '5678901234', @Salary1 = 65000

EXEC insertdata @Id1 = 106, @Name1 = 'nikhil', @Email1 = 'nikhil@gmail.com', @Mobile1 = '6789012345', @Salary1 = 59000

Output –

SELECT \* FROM employee



Que 2. Update using store procedure

Store procedure –

CREATE PROCEDURE update\_data(

@id1 AS INT,

@Name1 AS VARCHAR(50),

@Email1 AS VARCHAR(50),

@Mobile1 AS VARCHAR(50),

@Salary1 AS int

)

AS

BEGIN

UPDATE Employee SET Name = @Name1, Email = @Email1, Mobile = @Mobile1 , salary = @Salary1 WHERE Id = @id1;

END;

SQL query –

EXEC update\_data @id1 = 101, @Name1 = 'Mohan', @Email1 = 'Mohan@gmail.com', @Mobile1 = '9988776655', @Salary1 = 99999;

Output –

SELECT \* FROM Employee



* Update only Name

Store Procedure –

CREATE PROCEDURE update\_name(

@id1 AS INT,

@Name1 AS VARCHAR(50)

)

AS

BEGIN

UPDATE employee SET Name = @Name1 WHERE Id = @id1;

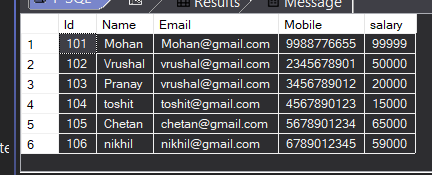
END;

SQL Query –

EXEC update\_name @id1 = 103, @Name1 = 'Pranay'

Output –

SELECT \* FROM Employee



Que 3. Store Procedure Delete.

Store Procedure –

CREATE PROCEDURE delete\_data(

@id1 AS INT

)

AS

BEGIN

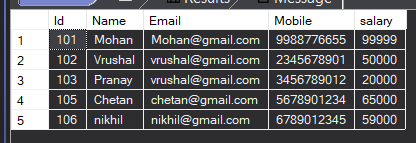
DELETE Employee WHERE Id = @id1;

END;

SQL Query –

EXEC delete\_data @id1 = 104;

Output –



Que 4 – Select data using store procedure

Store Procedure –

CREATE PROCEDURE rec\_select

AS

BEGIN

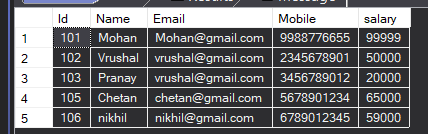
SELECT \* FROM Employee;

END;

SQL query –

EXEC rec\_select;

Output –



* Select specific record –

Store procedure –

CREATE PROCEDURE select\_rec(@Id1 AS INT)

AS

BEGIN

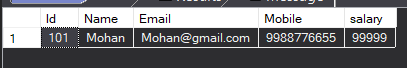
SELECT \* FROM Employee WHERE Id = @Id1;

END;

SQL Query –

EXEC select\_rec @Id1 = 101

Output –



Assignment 2 –

* Creating Actor Table

CREATE TABLE Actor (Act\_id INT PRIMARY KEY, Act\_fname VARCHAR(50), Act\_lname VARCHAR(50), Act\_gender VARCHAR(1))

INSERT INTO Actor VALUES

(101, 'Yash', 'Khan', 'M'),

(102, 'Vrushal', 'Ganar', 'M'),

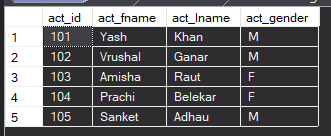
(103,'Amisha', 'Raut', 'F'),

(104, 'Prachi', 'Belekar', 'F'),

(105, 'Sanket', 'Adhau', 'M')

Output –

SELECT \* FROM Actor



* Create table director

CREATE TABLE Director(Dir\_id INT PRIMARY KEY, Dir\_fname VARCHAR(50), Dir\_lname VARCHAR(50))

INSERT INTO Director VALUES (201, 'Abhinav', 'Fasate'),

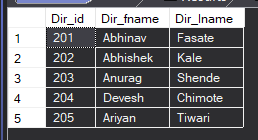
(202, 'Abhishek', 'Kale'),

(203, 'Anurag', 'Shende'),

(204, 'Devesh', 'Chimote'),

(205, 'Ariyan', 'Tiwari')

SELECT \* FROM Director;



* Create table Movie

CREATE TABLE Movie(Mov\_id INT PRIMARY KEY, Mov\_title VARCHAR(50), Mov\_year INT, Mov\_time INT, Mov\_lang VARCHAR(50),

mov\_dt\_rel DATE, mov\_rel\_country VARCHAR(50))

INSERT INTO Movie VALUES (301, 'Animal', 2023, 150, 'Hindi', '2023-12-01', 'India')

(302, 'Movie 1', 2022, 120, 'English', '2022-01-01', 'USA'),

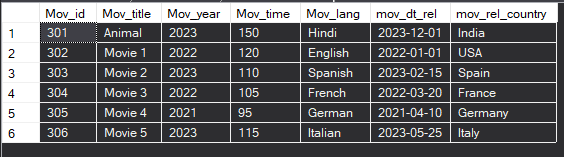
(303, 'Movie 2', 2023, 110, 'Spanish', '2023-02-15', 'Spain'),

(304, 'Movie 3', 2022, 105, 'French', '2022-03-20', 'France'),

(305, 'Movie 4', 2021, 95, 'German', '2021-04-10', 'Germany'),

(306, 'Movie 5', 2023, 115, 'Italian', '2023-05-25', 'Italy');

SELECT \* FROM Movie



* Create table reviewer

CREATE TABLE Reviewer(Rev\_id INT PRIMARY KEY, Rev\_name VARCHAR(50))

INSERT INTO Reviewer

VALUES

(401, 'Reviewer 1'),

(402, 'Reviewer 2'),

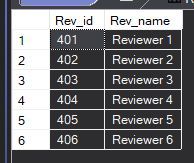
(403, 'Reviewer 3'),

(404, 'Reviewer 4'),

(405, 'Reviewer 5'),

(406, 'Reviewer 6');

SELECT \* FROM Reviewer;



* Create table Genres

CREATE TABLE Genres(Gen\_id INT PRIMARY KEY, Gen\_title VARCHAR(50));

INSERT INTO Genres

VALUES

(1, 'Genre 1'),

(2, 'Genre 2'),

(3, 'Genre 3'),

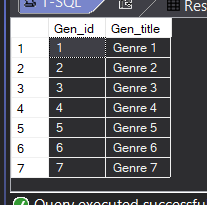
(4, 'Genre 4'),

(5, 'Genre 5'),

(6, 'Genre 6'),

(7, 'Genre 7');

SELECT \* FROM Genres;



* Create table Mov\_direction

CREATE TABLE Mov\_direction(Dir\_id INT NOT NULL, Mov\_id INT NOT NULL, FOREIGN KEY(Dir\_id) REFERENCES Director(Dir\_id))

INSERT INTO Mov\_direction

VALUES

(201, 301),

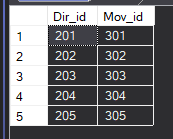
(202, 302),

(203, 303),

(204, 304),

(205, 305)

SELECT \* FROM Mov\_direction



* Create table Movie\_cast

CREATE TABLE Movie\_cast(Act\_id INT, Mov\_id INT,

FOREIGN KEY (Act\_id) REFERENCES Actor(Act\_id),

FOREIGN KEY(Mov\_id) REFERENCES Movie(Mov\_id)

)

INSERT INTO Movie\_cast

VALUES

(101, 301),

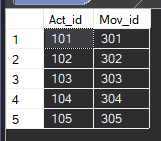
(102, 302),

(103, 303),

(104, 304),

(105, 305)

SELECT \* FROM Movie\_cast



* Create table Movie\_genres

CREATE TABLE movie\_genres (Mov\_id INT NOT NULL, Gen\_id INT NOT NULL

FOREIGN KEY (Mov\_id) REFERENCES Movie(Mov\_id),

FOREIGN KEY (Gen\_id) REFERENCES Genres(Gen\_id)

)

INSERT INTO movie\_genres

VALUES

(301, 1),

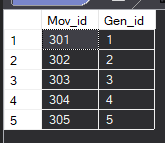
(302, 2),

(303, 3),

(304, 4),

(305, 5);

SELECT \* FROM movie\_genres



* Create Table rating

CREATE TABLE Rating (Mov\_id INT NOT NULL, Rev\_id INT NOT NULL, Rev\_stars INT, num\_o\_rating INT,

FOREIGN KEY (Mov\_id) REFERENCES Movie(Mov\_id),

FOREIGN KEY (Rev\_id) REFERENCES Reviewer

)

INSERT INTO Rating

VALUES

(301, 401, 4, 100),

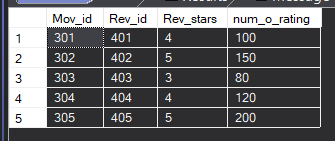
(302, 402, 5, 150),

(303, 403, 3, 80),

(304, 404, 4, 120),

(305, 405, 5, 200);

SELECT \* FROM Rating

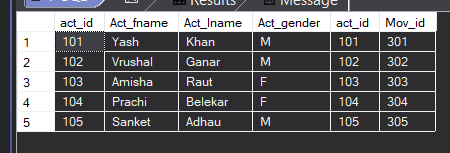


* Inner Join

SELECT Actor.act\_id, Actor.Act\_fname, Actor.Act\_lname, Actor.Act\_gender, Movie\_cast.act\_id, Movie\_cast.Mov\_id

FROM Actor

INNER JOIN Movie\_cast ON Actor.Act\_id = Movie\_cast.Act\_id

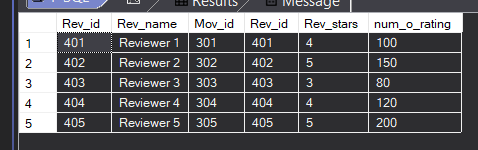


* Inner join

SELECT Reviewer.Rev\_id, Reviewer.Rev\_name, rating.Mov\_id, Rating.Rev\_id, Rating.Rev\_stars, Rating.num\_o\_rating

FROM Reviewer

INNER JOIN Rating ON Reviewer.Rev\_id = Rating.Rev\_id;

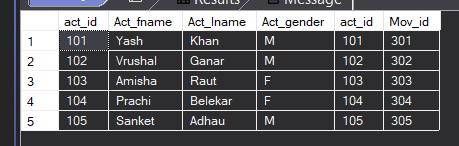


* Left Outer Join

SELECT Actor.act\_id, Actor.Act\_fname, Actor.Act\_lname, Actor.Act\_gender, Movie\_cast.act\_id, Movie\_cast.Mov\_id

FROM Actor

LEFT outer JOIN Movie\_cast ON Actor.Act\_id = Movie\_cast.Act\_id

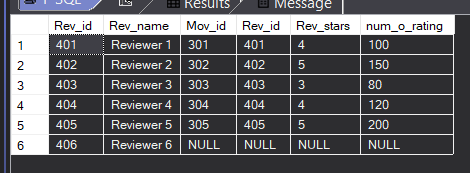


* Left outer join

SELECT Reviewer.Rev\_id, Reviewer.Rev\_name, rating.Mov\_id, Rating.Rev\_id, Rating.Rev\_stars, Rating.num\_o\_rating

FROM Reviewer

LEFT OUTER JOIN Rating ON Reviewer.Rev\_id = Rating.Rev\_id;

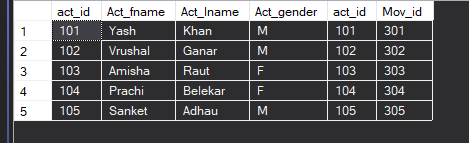


* Right Outer Join

SELECT Actor.act\_id, Actor.Act\_fname, Actor.Act\_lname, Actor.Act\_gender, Movie\_cast.act\_id, Movie\_cast.Mov\_id

FROM Actor

RIGHT OUTER JOIN Movie\_cast ON Actor.Act\_id = Movie\_cast.Act\_id

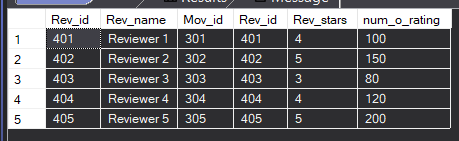


* Right Outer Join

SELECT Reviewer.Rev\_id, Reviewer.Rev\_name, rating.Mov\_id, Rating.Rev\_id, Rating.Rev\_stars, Rating.num\_o\_rating

FROM Reviewer

RIGHT OUTER JOIN Rating ON Reviewer.Rev\_id = Rating.Rev\_id;

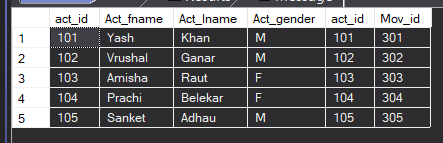


* Full Outer Join

SELECT Actor.act\_id, Actor.Act\_fname, Actor.Act\_lname, Actor.Act\_gender, Movie\_cast.act\_id, Movie\_cast.Mov\_id

FROM Actor

Full OUTER JOIN Movie\_cast ON Actor.Act\_id = Movie\_cast.Act\_id



* Full outer join

SELECT Reviewer.Rev\_id, Reviewer.Rev\_name, rating.Mov\_id, Rating.Rev\_id, Rating.Rev\_stars, Rating.num\_o\_rating

FROM Reviewer

FULL OUTER JOIN Rating ON Reviewer.Rev\_id = Rating.Rev\_id;

