# **Task1**

# **Project Title: Academic Management System (using SQL)**

**1. Database Creation**

CREATE DATABASE student\_database;

USE student\_database;

|  |  |
| --- | --- |
| **a)** | **b)** |
| *CREATE TABLE StudentInfo (*  *STU\_ID int PRIMARY KEY,*  *STU\_NAME varchar(100),*  *DOB DATE,*  *PHONE\_NO varchar(15),*  *EMAIL\_ID varchar(50),*  *ADDRESS varchar(250)*  *);* | CREATE TABLE CourseInfo (  COURSE\_ID INT,  COURSE\_NAME VARCHAR(100),  COURSE\_INSTRUCTOR\_NAME VARCHAR(100),  PRIMARY KEY (COURSE\_ID)  ); |
| **c)** |  |
| *CREATE TABLE EnrollmentInfo (*  *ENROLLMENT\_ID INT,*  *STU\_ID INT,*  *COURSE\_ID INT,*  *ENROLL\_STATUS VARCHAR(20),*  *PRIMARY KEY (ENROLLMENT\_ID),*  *FOREIGN KEY (STU\_ID) REFERENCES StudentInfo(STU\_ID),*  *FOREIGN KEY (COURSE\_ID) REFERENCES CourseInfo(COURSE\_ID)*  *);* |  |

**2. Data Creation:**

|  |
| --- |
|  |
| *INSERT INTO StudentInfo (STU\_ID, STU\_NAME, DOB, PHONE\_NO, EMAIL\_ID, ADDRESS) VALUES*  *(1001, 'Tom Hardy', '1993-08-23', '9999999991', 'tom101@gmail.com', 'Bangalore'),*  *(1002, 'Sam Joseph', '1994-08-23', '9999999992', 'sam102@gmail.com', 'Bangalore'),*  *(1003, 'Ben Issac', '1993-08-25', '9999999993', 'ben103@gmail.com', 'Chennai'),*  *(1004, 'Kane Lewis', '1993-10-23', '9999999994', 'kane104@gmail.com', 'Mumbai'),*  *(1005, 'Ian Robert', '1994-06-14', '9999999995', 'ian105@gmail.com', 'Delhi'),*  *(1006, 'John Austin', '1991-07-17', '9999999996', 'john106@gmail.com', 'Kochi');* |
|  |
| *INSERT INTO CourseInfo (COURSE\_ID, COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME) VALUES*  *(1, 'SQL', 'David'),*  *(2, 'Python', 'Artur'),*  *(3, 'AWS', 'Sebastian'),*  *(4, 'JAVA', 'Harry'),*  *(5, 'CSS', 'Jack');* |
|  |
| *INSERT INTO EnrollmentInfo (ENROLLMENT\_ID, STU\_ID, COURSE\_ID, ENROLL\_STATUS) VALUES*  *(10001, 1001, 1, 'ENROLLED'),*  *(10002, 1003, 2, 'ENROLLED'),*  *(10003, 1004, 4, 'ENROLLED'),*  *(10004, 1002, 3, 'ENROLLED'),*  *(10005, 1005, 3, 'NOT ENROLLED'),*  *(10006, 1006, 5, 'ENROLLED');* |

3) **Retrieve the Student Information**

|  |  |
| --- | --- |
| a) Write a query to retrieve student details, such as student name, contact informations, and Enrollment status. | |
| SELECT s.STU\_NAME, s.PHONE\_NO, s.ADDRESS, e.ENROLL\_STATUS  FROM StudentInfo s  JOIN EnrollmentInfo e  ON s.STU\_ID = e.STU\_ID  ORDER BY e.ENROLL\_STATUS ASC; |  |
|  |  |
| b) Write a query to retrieve a list of courses in which a specific student is enrolled. | |
| *SELECT c.COURSE\_NAME, s.STU\_NAME*  *FROM EnrollmentInfo e*  *JOIN CourseInfo c ON e.COURSE\_ID = c.COURSE\_ID*  *JOIN StudentInfo s ON s.STU\_ID = e.STU\_ID;* |  |
|  |  |
| c) Write a query to retrieve course information, including course name, instructor information. | |
| *SELECT COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME*  *FROM CourseInfo;* |  |
|  | |
| d) Write a query to retrieve course information for a specific course . | |
| *SELECT COURSE\_ID, COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME*  *FROM CourseInfo*  *WHERE COURSE\_NAME = 'SQL';* |  |
|  |  |
| e) Write a query to retrieve course information for multiple courses. | |
| *SELECT COURSE\_ID, COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME*  *FROM CourseInfo*  *WHERE COURSE\_NAME IN ('SQL', 'Python');* |  |
|  |  |
| f) Test the queries to ensure accurate retrieval of student information. (Execute the queries and verify the results against the expected output.) | |
| *SELECT \* FROM StudentInfo;* |  |

**4. Reporting and Analytics (Using joining queries)**

|  |  |
| --- | --- |
| a) Write a query to retrieve the number of students enrolled in each course | |
| *SELECT c.Course\_Name, COUNT(e.course\_id) AS numberofStud*  *FROM CourseInfo c*  *JOIN EnrollmentInfo e ON c.course\_id = e.course\_ID*  *WHERE e.enroll\_status = 'ENROLLED'*  *GROUP BY c.Course\_Name;* |  |
|  |  |
| b) Write a query to retrieve the list of students enrolled in a specific course | |
| *SELECT e.COURSE\_ID, c.COURSE\_NAME, s.STU\_NAME*  *FROM CourseInfo c*  *JOIN EnrollmentInfo e ON c.course\_id = e.course\_ID*  *JOIN StudentInfo s ON s.STU\_ID = e.STU\_ID*  *WHERE e.enroll\_status = 'ENROLLED';* |  |
|  |  |
| c) Write a query to retrieve the count of enrolled students for each instructor. | |
| *SELECT c.COURSE\_INSTRUCTOR\_NAME, COUNT(e.STU\_ID) AS numberofStud*  *FROM CourseInfo c*  *JOIN EnrollmentInfo e ON c.course\_id = e.course\_ID*  *WHERE e.enroll\_status = 'ENROLLED'*  *GROUP BY c.COURSE\_INSTRUCTOR\_NAME;* |  |
|  |  |
| d) Write a query to retrieve the list of students who are enrolled in multiple courses | |
| *SELECT e.stu\_id, COUNT(c.course\_id) AS numberofStud*  *FROM CourseInfo c*  *JOIN EnrollmentInfo e ON c.course\_id = e.course\_ID*  *WHERE e.enroll\_status = 'ENROLLED'*  *GROUP BY e.stu\_id*  *HAVING COUNT(c.course\_id) > 1;* |  |
|  |  |
| e) Write a query to retrieve the courses that have the highest number of enrolled students (arranging from highest to lowest) | |
| *SELECT c.COURSE\_ID, c.COURSE\_NAME, COUNT(e.STU\_ID) AS numberofStud*  *FROM CourseInfo c*  *JOIN EnrollmentInfo e ON c.course\_id = e.course\_ID*  *WHERE e.enroll\_status = 'ENROLLED'*  *GROUP BY c.COURSE\_ID, c.COURSE\_NAME*  *ORDER BY COUNT(e.STU\_ID) DESC;* |  |
|  |  |

# **Task2**

# **Project: Student Database Management System(PostgreSQL)**

1. **Database setup**

CREATE DATABASE student\_database

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'English\_United States.1252'

LC\_CTYPE = 'English\_United States.1252'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1;

|  |  |
| --- | --- |
| a) Create student database: | b) Student table creation: |
| *CREATE DATABASE student\_database*  *WITH*  *OWNER = postgres*  *ENCODING = 'UTF8'*  *LC\_COLLATE = 'English\_United States.1252'*  *LC\_CTYPE = 'English\_United States.1252'*  *TABLESPACE = pg\_default*  *CONNECTION LIMIT = -1;* | *CREATE TABLE Student\_table (*  *Student\_id INT,*  *Stu\_name VARCHAR(100),*  *Department VARCHAR(50),*  *Email\_id VARCHAR(50),*  *Phone\_no NUMERIC,*  *Address VARCHAR(250),*  *Date\_Of\_Birth DATE,*  *Gender VARCHAR(30),*  *Major VARCHAR(50),*  *GPA NUMERIC,*  *Grade VARCHAR(10)*  *);* |
|  |  |

**2) Data entry**

INSERT INTO Student\_table (Student\_id, Stu\_name, Department, email\_id, Phone\_no, Address, Date\_Of\_Birth, Gender, Major, GPA, Grade)

VALUES

('1', 'Muskaan Arya', 'Business', 'muskaan@gmail.com', '9999999991', 'Delhi', '1999-04-26', 'Female', 'MBA', '8.8', 'A'),

('2', 'Kundan Kumar', 'Arts and Sciences', 'kundan@gmail.com', '9999999992', 'Bangalore', '1992-07-15', 'Male', 'Mathematics', '8.6', 'A'),

('3', 'Rajat Nema', 'Business', 'Rajat@gmail.com', '9999999993', 'Delhi', '1995-06-28', 'Male', 'MBA', '8.5', 'A'),

('4', 'Devashish Negi', 'Arts and Sciences', 'Devashish@gmail.com', '9999999994', 'Dehradhun', '1997-01-12', 'Male', 'Physics', '7.6', 'B'),

('5', 'Karishma Roy', 'Arts and communication', 'Karishma@gmail.com', '9999999995', 'Mumbai', '1995-10-26', 'Female', 'Communication', '6.9', 'B'),

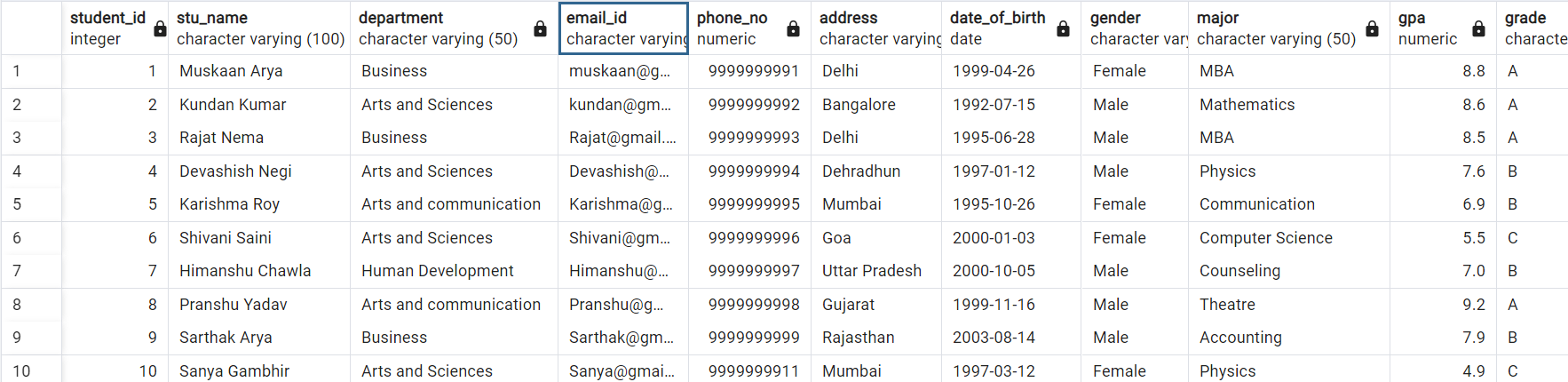
('6', 'Shivani Saini', 'Arts and Sciences', 'Shivani@gmail.com', '9999999996', 'Goa', '2000-01-03', 'Female', 'Computer Science', '5.5', 'C'),

('7', 'Himanshu Chawla', 'Human Development', 'Himanshu@gmail.com', '9999999997', 'Uttar Pradesh', '2000-10-05', 'Male', 'Counseling', '7.0', 'B'),

('8', 'Pranshu Yadav', 'Arts and communication', 'Pranshu@gmail.com', '9999999998', 'Gujarat', '1999-11-16', 'Male', 'Theatre', '9.2', 'A'),

('9', 'Sarthak Arya', 'Business', 'Sarthak@gmail.com', '9999999999', 'Rajasthan', '2003-08-14', 'Male', 'Accounting', '7.9', 'B'),

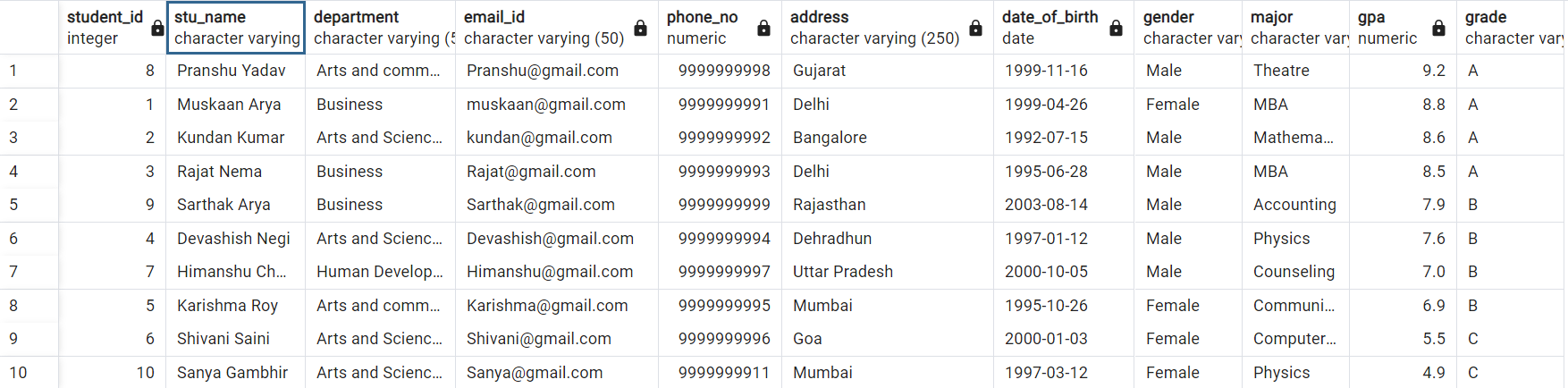
('10', 'Sanya Gambhir', 'Arts and Sciences', 'Sanya@gmail.com', '9999999911', 'Mumbai', '1997-03-12', 'Female', 'Physics', '4.9', 'C');



**3) Student information retrieval**

SELECT \*

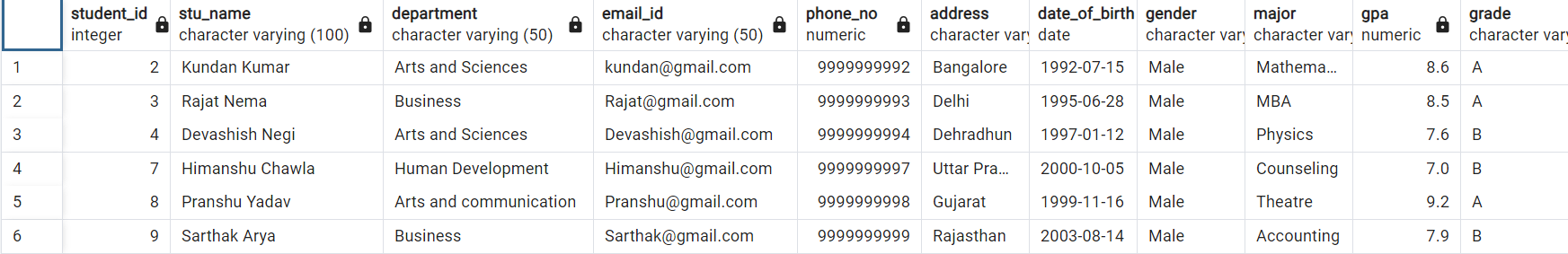
FROM Student\_table

ORDER BY GPA DESC, Grade;

**4) Query for Male students**

Select \* from Student\_table

where Gender = 'Male';

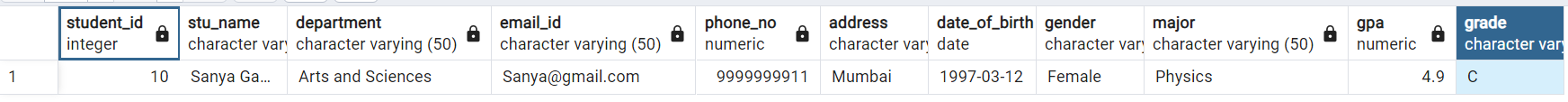


**5)GPA less than 5**

SELECT \*

FROM Student\_table

WHERE GPA <5.0;



**6)Update email\_id and Grade**

UPDATE Student\_table

SET email\_id = 'Sanyaa@gmail.com', Grade = 'D'

WHERE Student\_id = '10';

**7)Query Grade B**

SELECT Stu\_name, date\_part('year',age(Date\_Of\_Birth)) as Age

FROM Student\_table

Where Grade = 'B';

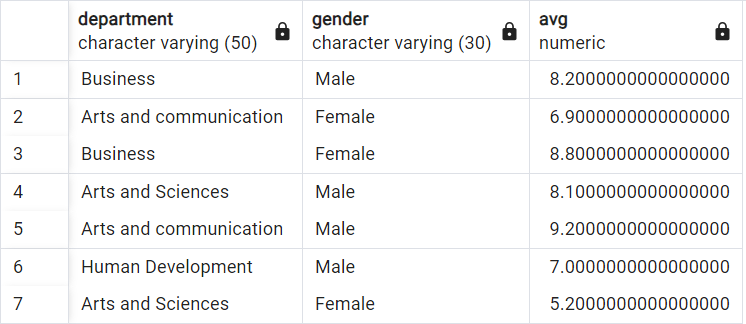


**8) Grouping and calculation**

SELECT Department, Gender, Avg(GPA)

FROM Student\_table

GROUP BY 1,2;

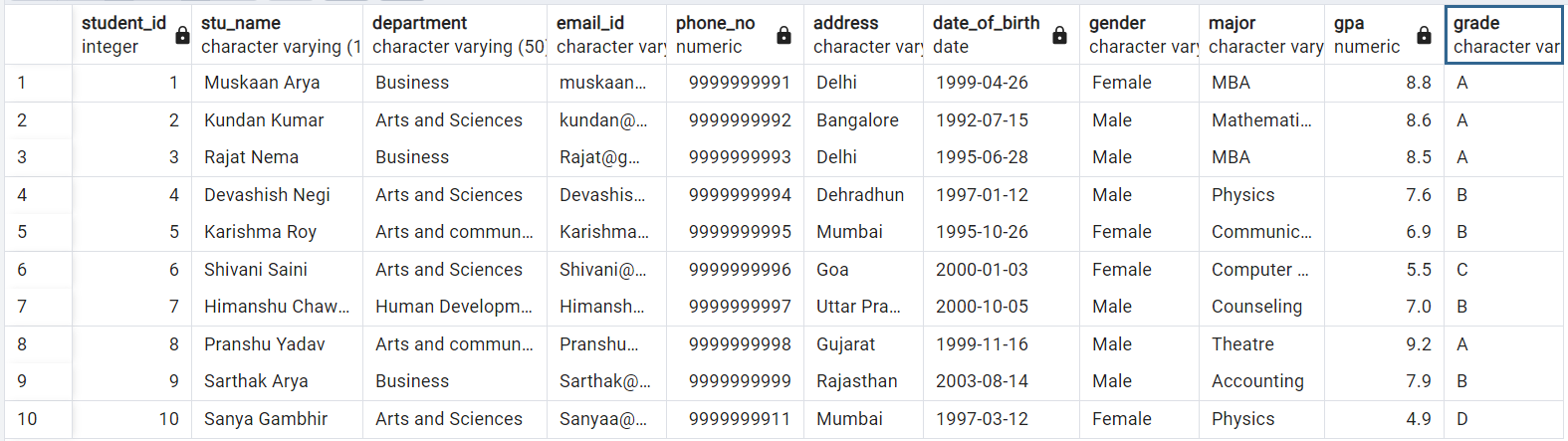


**9)Table Renaming**

ALTER TABLE Student\_table

RENAME TO Student\_info;

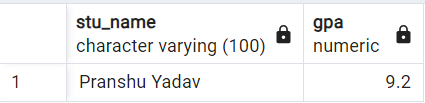
SELECT \* FROM Student\_info;



**10)Students with highest GPA**

SELECT Stu\_name, GPA

FROM Student\_info WHERE GPA = (Select Max(GPA) From Student\_info);



# **Task 3**

# **Project: Event Management System using PostgreSQL.**

1. **Database creation**

CREATE DATABASE "EventsManagement"

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'English\_United States.1252'

LC\_CTYPE = 'English\_United States.1252'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

1. Table Creation

|  |  |
| --- | --- |
| Events | Attendees |
| *CREATE TABLE Events (*  *Event\_Id INT,*  *Event\_Name VARCHAR(30),*  *Event\_Date DATE,*  *Event\_Location VARCHAR(100),*  *Event\_Description VARCHAR(200),*  *PRIMARY KEY (Event\_Id)*  *);* | *CREATE TABLE Attendees (*  *Attendee\_Id INT,*  *Attendee\_Name VARCHAR(30),*  *Attendee\_Phone NUMERIC,*  *Attendee\_Email VARCHAR(30),*  *Attendee\_City VARCHAR(20),*  *PRIMARY KEY (Attendee\_Id)*  *);* |
|  |  |
| Registrations |  |
| *CREATE TABLE Registrations (*  *Registration\_Id INT,*  *Event\_Id INT,*  *Attendee\_Id INT,*  *Registration\_Date DATE,*  *Registration\_Amount NUMERIC,*  *PRIMARY KEY (Registration\_Id),*  *FOREIGN KEY (Event\_Id) REFERENCES Events(Event\_Id),*  *FOREIGN KEY (Attendee\_Id) REFERENCES Attendees(Attendee\_Id)*  *);* |  |

**2. Data Creation**

INSERT INTO Events

(Event\_Id, Event\_Name, Event\_Date, Event\_Location, Event\_Description) VALUES

('101', 'Ed Sheeran', '2024-03-03', 'Mumbai', 'Music Show'),

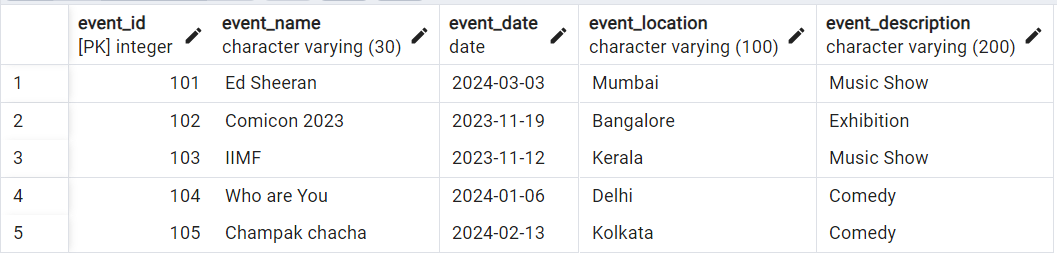
('102', 'Comicon 2023', '2023-11-19', 'Bangalore', 'Exhibition'),

('103', 'IIMF', '2023-11-12', 'Kerala', 'Music Show'),

('104', 'Who are You', '2024-01-06', 'Delhi', 'Comedy'),

('105', 'Champak chacha', '2024-02-13', 'Kolkata', 'Comedy');

SELECT \* FROM Events;



INSERT INTO Attendees (Attendee\_Id, Attendee\_Name, Attendee\_Phone, Attendee\_Email, Attendee\_City) VALUES

('1001', 'Muskaan', '9999999991', 'Muskaan@gmail.com', 'Delhi'),

('1002', 'Rajat', '9999999992', 'Rajat@gmail.com', 'Delhi'),

('1003', 'Abhinav', '9999999993', 'Abhinav@gmail.com', 'Mumbai'),

('1004', 'Kundan', '9999999994', 'Kundan@gmail.com', 'Uttar Pradesh'),

('1005', 'Karishma', '9999999995', 'Karishma@gmail.com', 'Mumbai'),

('1006', 'Shivani', '9999999996', 'Shivani@gmail.com', 'Goa'),

('1007', 'Devashish', '9999999997', 'Devashish@gmail.com', 'Uttrakhand'),

('1008', 'Sarthak', '9999999998', 'Sarthak@gmail.com', 'Delhi');

SELECT \* FROM Attendees;



INSERT INTO Registrations

(Registration\_Id, Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount) VALUES

('10001', '101', '1001', '2023-10-12', '7500'),

('10002', '102', '1003', '2023-09-25', '1000'),

('10003', '103', '1002', '2023-10-29', '2000'),

('10004', '104', '1004', '2023-10-20', '500'),

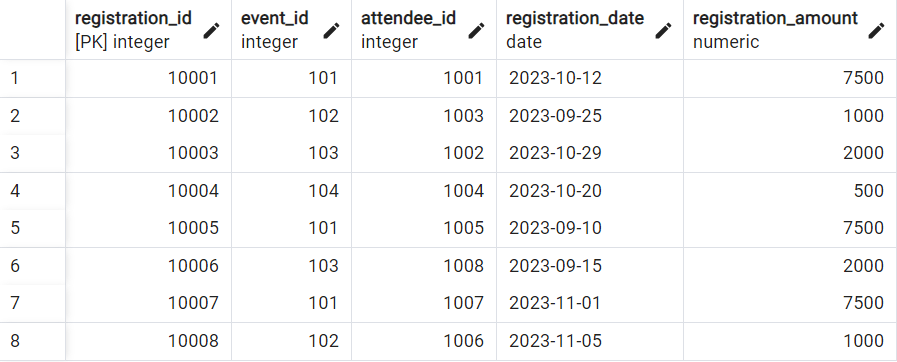
('10005', '101', '1005', '2023-09-10', '7500'),

('10006', '103', '1008', '2023-09-15', '2000'),

('10007', '101', '1007', '2023-11-01', '7500'),

('10008', '102', '1006', '2023-11-05', '1000');

SELECT \* FROM Registrations;



**3) Manage Event Details**

a) Insert New event

Insert INTO Events (Event\_Id, Event\_Name, Event\_Date, Event\_Location, Event\_Description) VALUES

('106', 'Unheard Diaries', '2023-10-29', 'Delhi', 'Storytelling');

b) Update event's information

UPDATE Events

SET Event\_Location = 'Bangalore' WHERE Event\_Id = '104';

c) Deleting an event

DELETE FROM Events WHERE Event\_Id = '105';

**4) Manage Track attendees and handle events**

a) Insert new attendee

INSERT INTO Attendees (Attendee\_Id, Attendee\_Name, Attendee\_Phone, Attendee\_Email, Attendee\_City)

VALUES ('1009', 'Krishna', '9999999999', 'Krishna@gmail.com', 'Tamil Nadu');

b) Register attendee

INSERT INTO Registrations

(Registration\_Id, Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount)

VALUES ('10009', '101', '1009', '2023-11-11', '7500');

**5) Retrieve event information, generate attendee list, Calculate event attendee statistics**

WITH Event1 AS(

SELECT E.Event\_id, E.event\_name, E.event\_date, E.event\_location,

SUM(R.registration\_amount) OVER(PARTITION by E.event\_id) AS Amountgenperevent

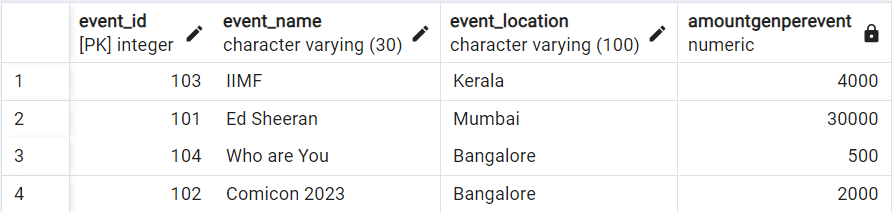
FROM Events E JOIN Registrations R ON E.event\_id = R.event\_id

JOIN Attendees A ON A.attendee\_id = R.attendee\_id)

SELECT Event\_id, event\_name, event\_location, Amountgenperevent

FROM Event1

GROUP BY 1,2,3,4;



# **Task4.**

# **Project: OLAP Operations (using Redshift or PostgreSQL)**

*CREATE DATABASE "Sales Data "*

*WITH*

*OWNER = postgres*

*ENCODING = 'UTF8'*

*LC\_COLLATE = 'English\_United States.1252'*

*LC\_CTYPE = 'English\_United States.1252'*

*TABLESPACE = pg\_default*

*CONNECTION LIMIT = -1*

*IS\_TEMPLATE = False;*

**1) Database creation**

CREATE TABLE Sales\_sample (Product\_Id INT, Region VARCHAR(50), On\_date DATE,

Sales\_Amount NUMERIC);

**2) Data Creation**

|  |  |
| --- | --- |
| INSERT INTO Sales\_sample (Product\_Id, Region, On\_date, Sales\_Amount) VALUES  ('1', 'East', '2023-10-10', '20000'),  ('2', 'West', '2023-09-19', '50000'),  ('2', 'East', '2023-10-21', '40000'),  ('3', 'North', '2023-09-20', '15000'),  ('4', 'North', '2023-08-06', '45000'),  ('2', 'South', '2023-08-25', '45000'),  ('5', 'North', '2023-11-23', '20000'),  ('5', 'West', '2023-11-11', '60000'),  ('3', 'East', '2023-09-19', '50000'),  ('1', 'West', '2023-09-29', '70000');  Select \* from Sales\_Sample; |  |

**3) OLAP operations**

a) Drill down

SELECT Region, Product\_Id, Sum(Sales\_Amount) AS Sales\_Amount

FROM Sales\_Sample

GROUP BY 1,2

ORDER BY Region, Product\_Id, Sales\_Amount;



b) Roll Up

|  |  |
| --- | --- |
| *SELCT Region, Product\_Id, Sum(Sales\_Amount) AS Sales\_Amount*  *FROM Sales\_Sample*  *GROUP BY ROLLUP (1,2)*  *ORDER BY Region;* |  |

c) Cube

|  |  |
| --- | --- |
| *SELECT Region, Product\_Id, On\_Date, SUM(Sales\_Amount) AS Sales\_Amount*  *FROM Sales\_Sample*  *GROUP BY Cube (1,2,3)*  *ORDER BY Region, Product\_Id, On\_Date, Sales\_Amount;* |  |

d) Slice

|  |  |
| --- | --- |
| *SELECT Region, Product\_Id, On\_Date, SUM(Sales\_Amount) AS Sales\_Amount*  *FROM Sales\_Sample*  *WHERE Region in('North', 'South') OR On\_Date BETWEEN To\_date('2023-08-20','YYYY-MM-DD') AND To\_Date('2023-10-20','YYYY-MM-DD')*  *GROUP BY 1,2,3*  *ORDER BY Region, Product\_Id, On\_Date, Sales\_Amount;* |  |

e) Dice

|  |  |
| --- | --- |
| *SELECT Region, Product\_Id, On\_Date, SUM(Sales\_Amount) AS Sales\_Amount*  *FROM Sales\_Sample*  *WHERE Region in('North', 'South') AND Product\_Id IN (1,2) AND On\_Date BETWEEN To\_date('2023-08-20','YYYY-MM-DD') And To\_Date('2023-10-20','YYYY-MM-DD')*  *GROUP BY 1,2,3*  *ORDER BY Region, Product\_Id, On\_Date, Sales\_Amount;* |  |