**CAPSTONE PROJECT**

**Task 1 (Workbench)**

**1) Database Creation**

create database student\_database;

use student\_database;

1. **create table StudentInfo**

(STU\_ID int, STU\_NAME varchar(100),

DOB DATE, PHONE\_NO VARCHAR(10),EMAIL\_ID varchar(50),

ADDRESS varchar(250),primary key (STU\_ID));

1. **create table CourseInfo**

(COURSE\_ID INT ,COURSE\_NAME VARCHAR(100), COURSE\_INSTRUCTOR\_NAME VARCHAR(100), primary key (COURSE\_ID));

1. **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**

a) Insert Into StudentInfo

(STU\_ID, STU\_NAME, DOB, PHONE\_NO, EMAIL\_ID ,ADDRESS) Values

('1001', 'Tom Hardy', '1993-08-23',9999999991, 'tom101@gamil.com', 'Banglore'),

('1002', 'Sam Joseph', '1994-08-23',9999999992, 'sam102@gamil.com', 'Banglore'),

('1003', 'Ben Issac', '1993-08-25',9999999993, 'ben103@gamil.com', 'Chennai'),

('1004', 'Kane Lewis', '1993-10-23',9999999994, 'kane104@gamil.com', 'Mumbai'),

('1005', 'Ian Robert', '1994-06-14',9999999995, 'ian105@gamil.com', 'Delhi'),

('1006', 'John Austin', '1991-07-17',9999999996, 'john106@gamil.com', 'Kochi');

b) 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');

c) insert into EnrollmentInfo(ENROLLMENT\_ID, STU\_ID, COURSE\_ID, ENROLL\_STATUS) values

(10001, 1001, 001,'ENROLLED'),

(10002, 1003, 002,'ENROLLED'),

(10003, 1004, 004,'ENROLLED'),

(10004, 1002, 003,'ENROLLED'),

(10005, 1005, 003,'NOT ENROLLED'),

(10006, 1006, 005,'ENROLLED');

3) **Retrieve the Student Information**

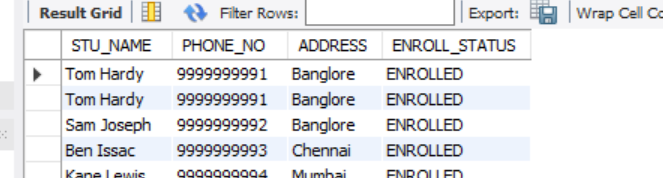
**a) Write a query to retrieve Student details, Such as Student Name, Contact Informations and Enrollment Status**

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;

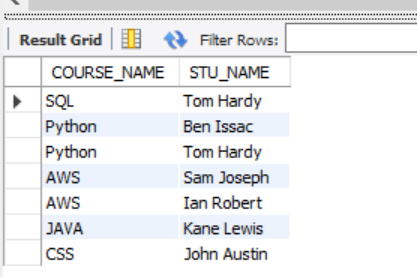


**b) Write a query to retrieve a list of courses in which a specific student enrolled**

select c.COURSE\_NAME, s.STU\_NAME

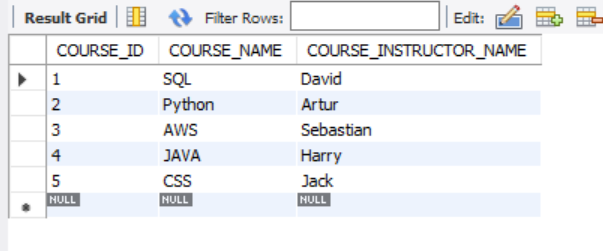
from EnrollmentInfo e join CourseInfo c on e.course\_id = c.course\_id

join StudentInfo s on s.STU\_ID = e.STU\_ID



**C) Write a query to retrieve course information, including course name, insturctor information**

Select \* From CourseInfo;

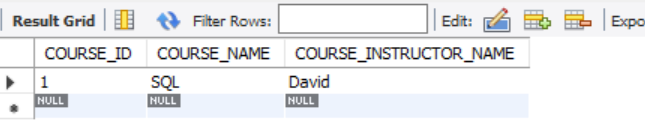


**d) Write a query to retrieve course information for a specific course**

Select COURSE\_ID,COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME

from CourseInfo

where COURSE\_NAME='SQL';

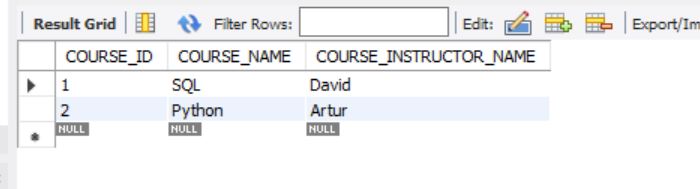


**e) Write a query to retrieve course information for multiple courses**

Select COURSE\_ID,COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME

from CourseInfo

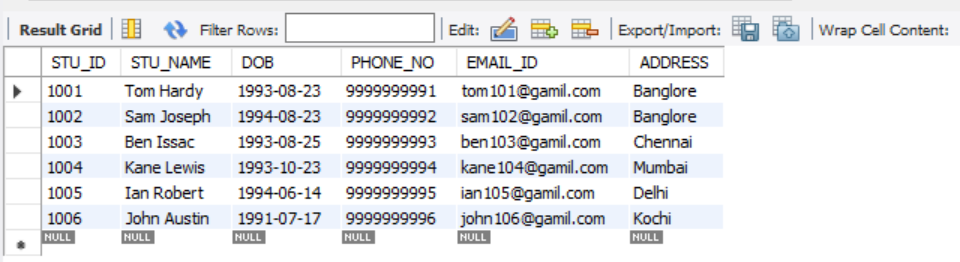
where COURSE\_NAME in ('SQL','Python');



**f) Test the queries to ensure accurate retrieval of Student Information**

**(Execute 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**

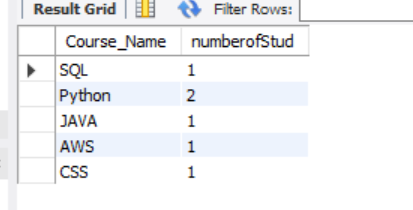
Select c.Course\_Name , count(c.course\_id) as numberofStud

from CourseInfo c join EnrollmentInfo e

on c.course\_id=e.course\_ID

where e.enroll\_status = 'ENROLLED'

group by 1

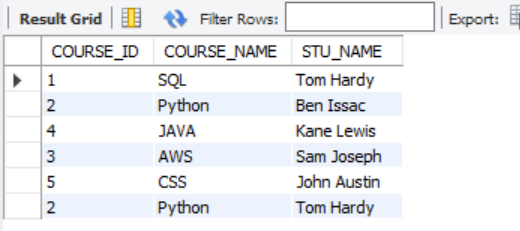


**b) Write a query to retrieve the list of students enrolled in a specific course**

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**

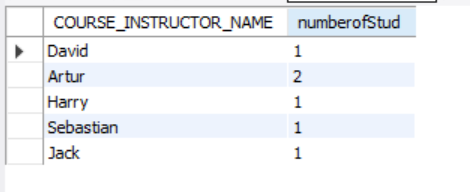
Select c.COURSE\_INSTRUCTOR\_NAME , count(e.Stu\_id) as numberofStud

from CourseInfo c join EnrollmentInfo e

on c.course\_id=e.course\_ID

where e.enroll\_status = 'ENROLLED'

group by 1;



**d) Write a query to retrieve the list of students who are enrolled in a multiple course**

Select e.stu\_id , count(c.course\_id) as numberofStud

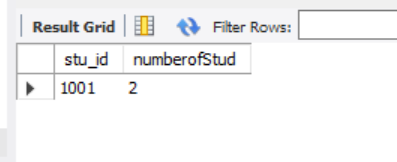
from CourseInfo c join EnrollmentInfo e

on c.course\_id=e.course\_ID

where e.enroll\_status = 'ENROLLED'

group by 1

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 e.stu\_id , count(c.course\_id) as numberofStud

from CourseInfo c join EnrollmentInfo e

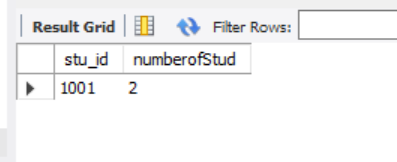
on c.course\_id=e.course\_ID

where e.enroll\_status = 'ENROLLED'

group by 1

having count(c.course\_id) >1

order by count(c.course\_id) desc



**Task 2 (Postgresql)**

-- Database: Student\_Database

-- DROP DATABASE IF EXISTS "Student\_Database";

**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

IS\_TEMPLATE = False;

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'),

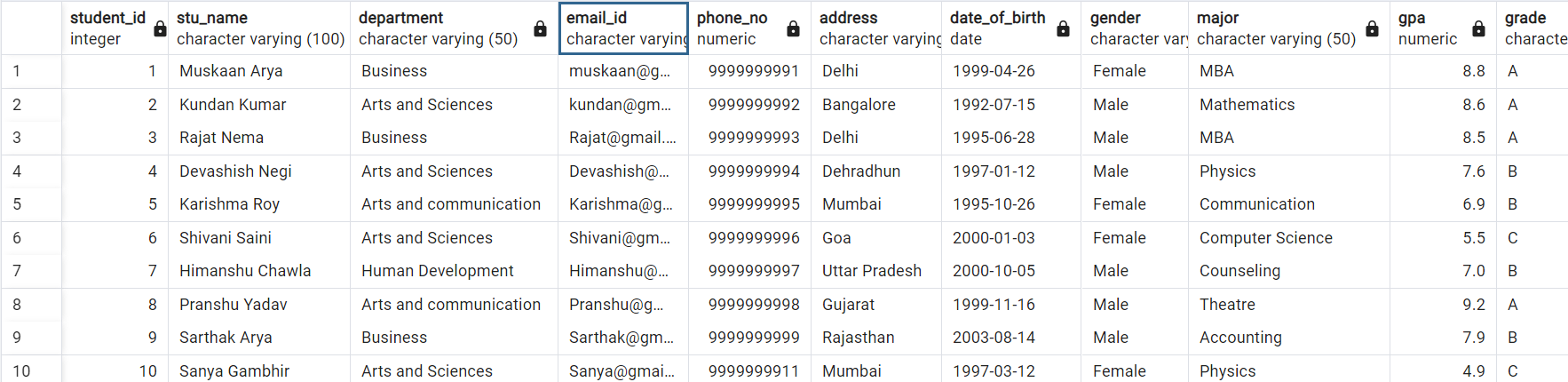
('7', 'Himanshu Chawla', 'Human Development', 'Himanshu@gmail.com', '9999999997', 'Uttar Pradesh', '2000-10-05', 'Male', 'Counseling', '7.0', 'B'),

('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');

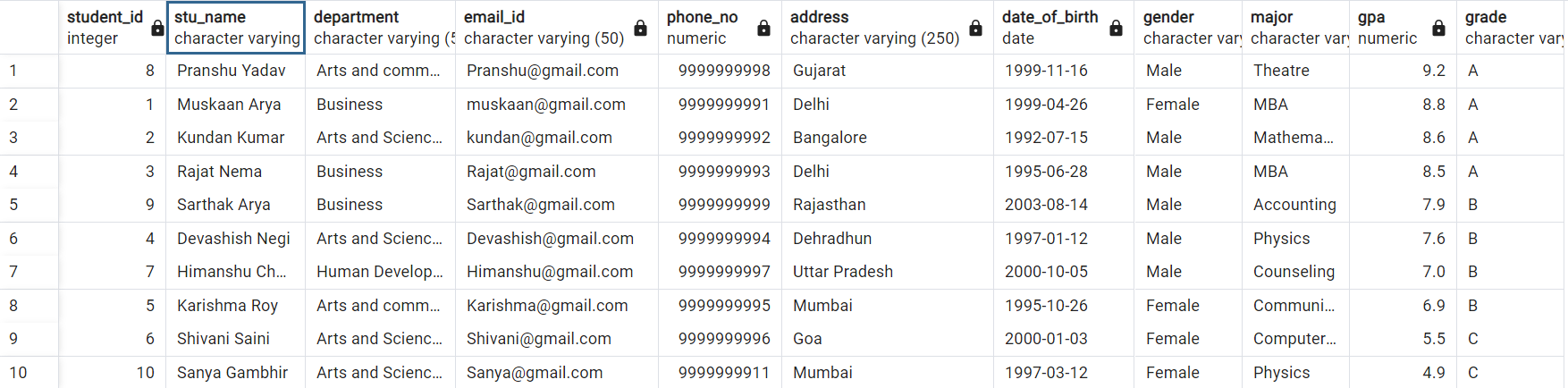
Select \* from Student\_table;



**3) Student information retrieval**

Select \* from Student\_table

Order By GPA desc, Grade;

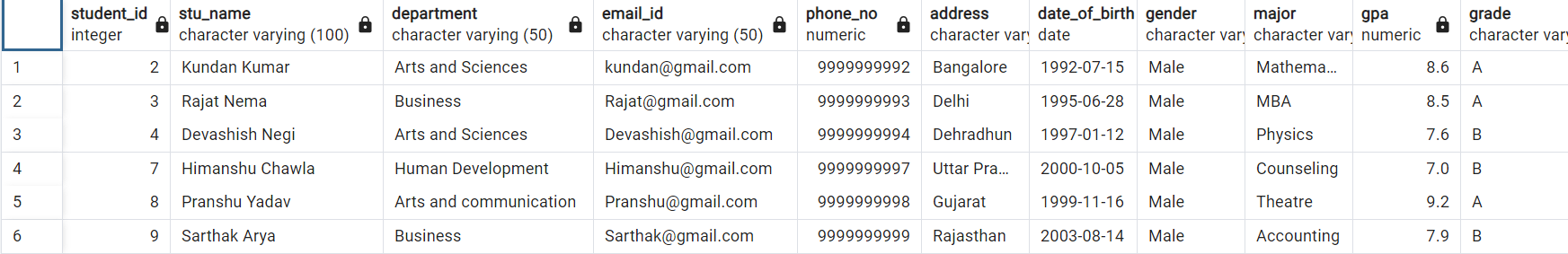


**4) Male students**

Select \* from Student\_table

where Gender = 'Male'

;

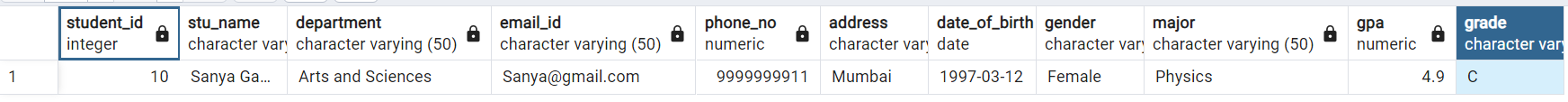


**5) GPA less than 5**

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) Grade B**

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

FROM Student\_table

Where Grade = 'B'

;



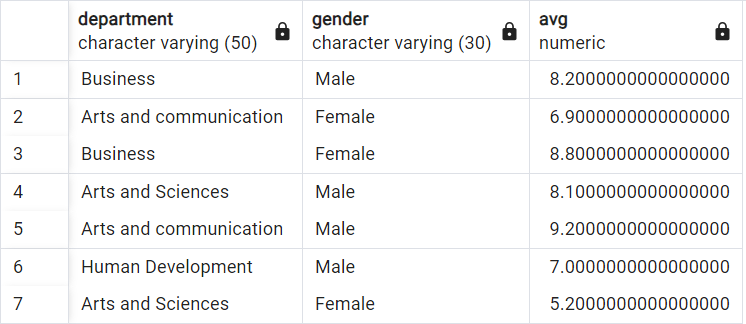
**8) Grouping and calculation**

Select Department, Gender, Avg(GPA)

From Student\_table

Group By 1,2

;

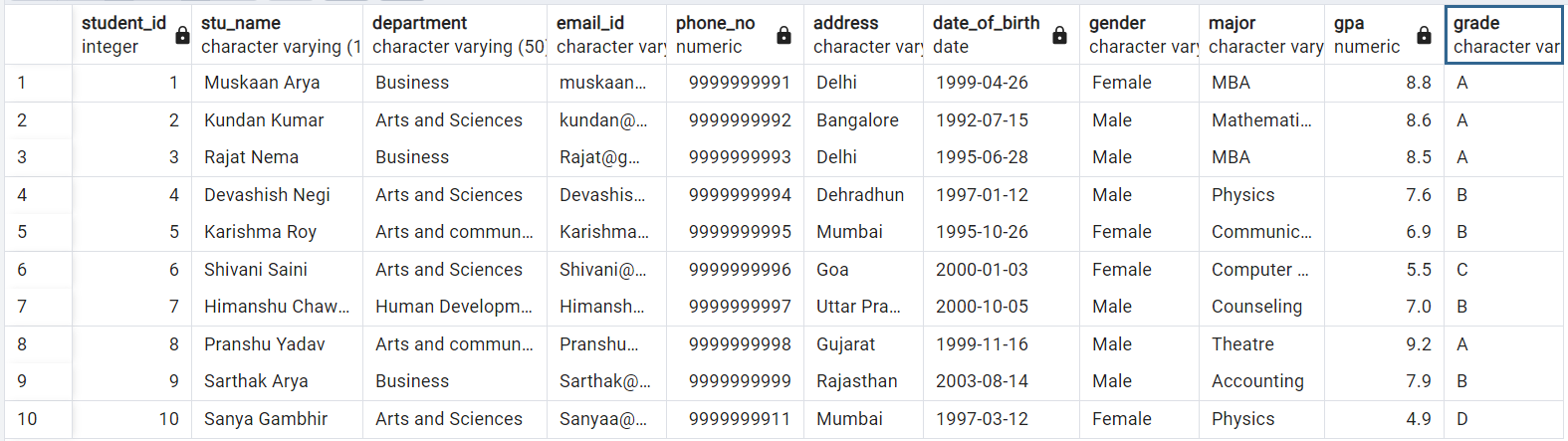


**9) Renaming**

Alter table Student\_table

Rename to Student\_info;

Select \* from Student\_info

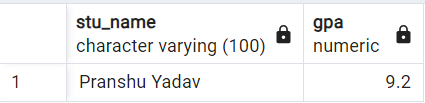


**10) Students with highest GPA**

Select Stu\_name, GPA

From Student\_info where GPA = (Select Max(GPA) From Student\_info)

;



**Task 3 (Postgresql)**

-- Database: EventsManagement

-- DROP DATABASE IF EXISTS "EventsManagement";

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) Database Creation**

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));

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'),

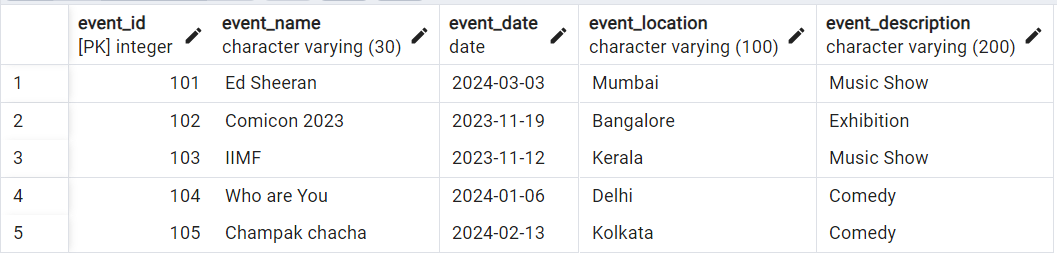
('103', 'IIMF', '2023-11-12', 'Kerala', 'Music Show'),

('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'),

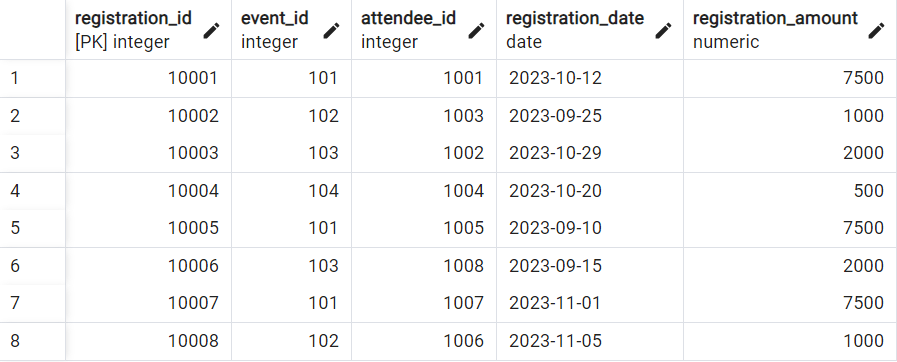
('10006', '103', '1008', '2023-09-15', '2000'),

('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**

Select \* from Events;

Select \* from Attendees;

select \* from Registrations;

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

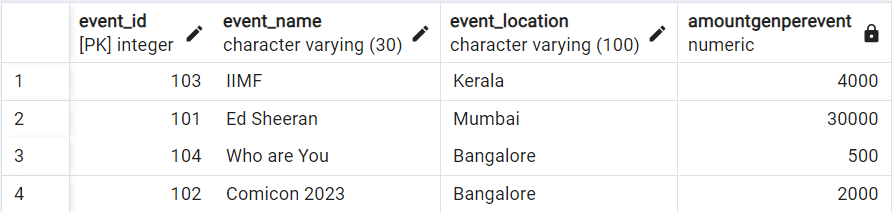
from Events E join Registrations R on E.event\_id = R.event\_id

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



**Task 4 (Postgresql)**

-- Database: Sales Data

-- DROP DATABASE IF EXISTS "Sales Data ";

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'),

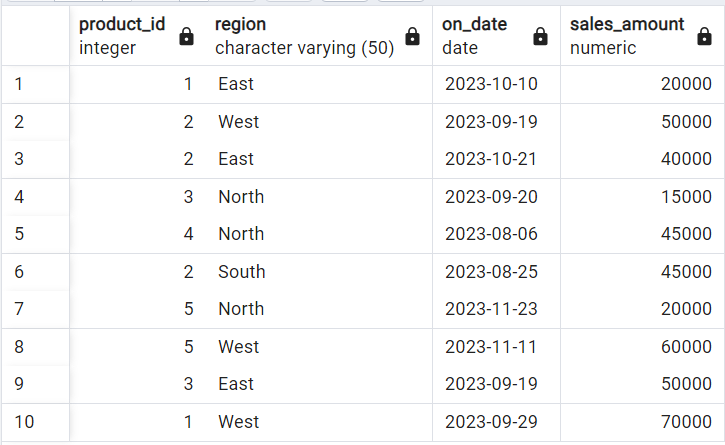
('5', 'West', '2023-11-11', '60000'),

('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**

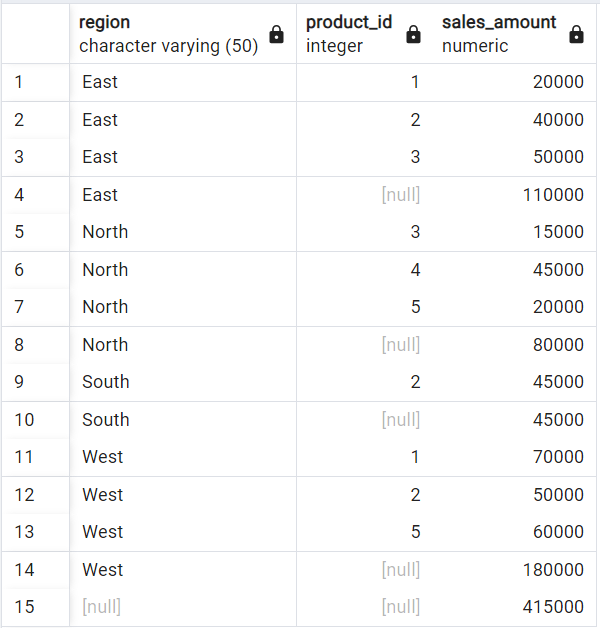
Select Region, Product\_Id, Sum(Sales\_Amount) as Sales\_Amount

From Sales\_Sample

Group By Rollup (1,2)

Order By Region

;



**c) Cube**

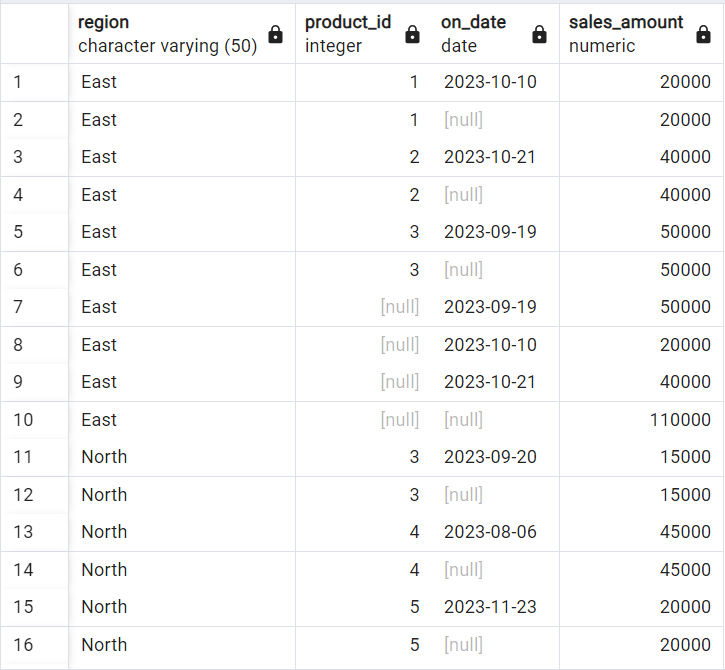
Select Region, Product\_Id, On\_Date, Sum(Sales\_Amount) as Sales\_Amount

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

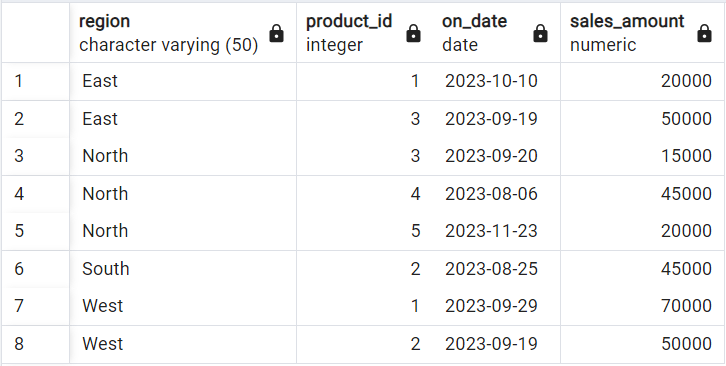
From Sales\_Sample

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

;

