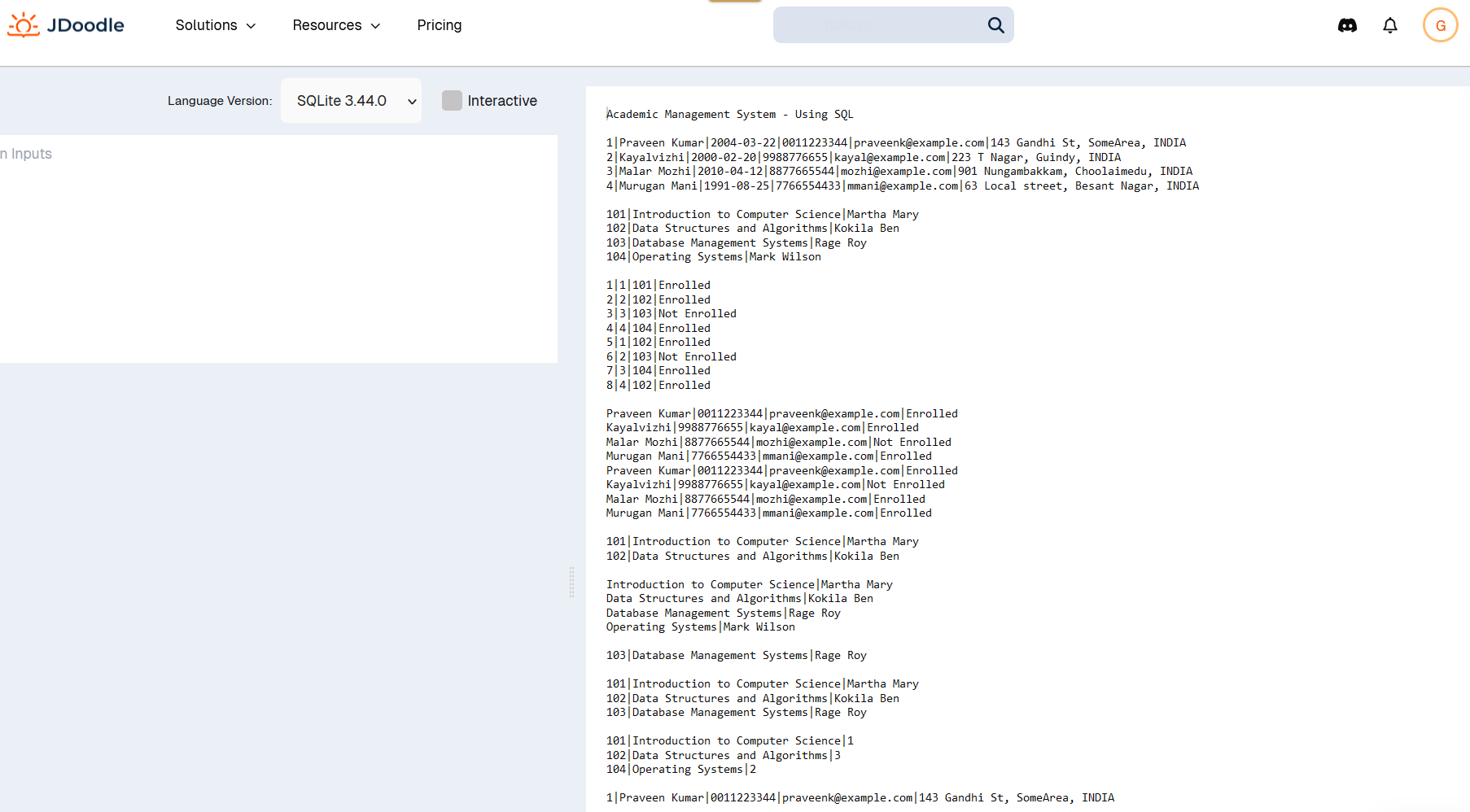
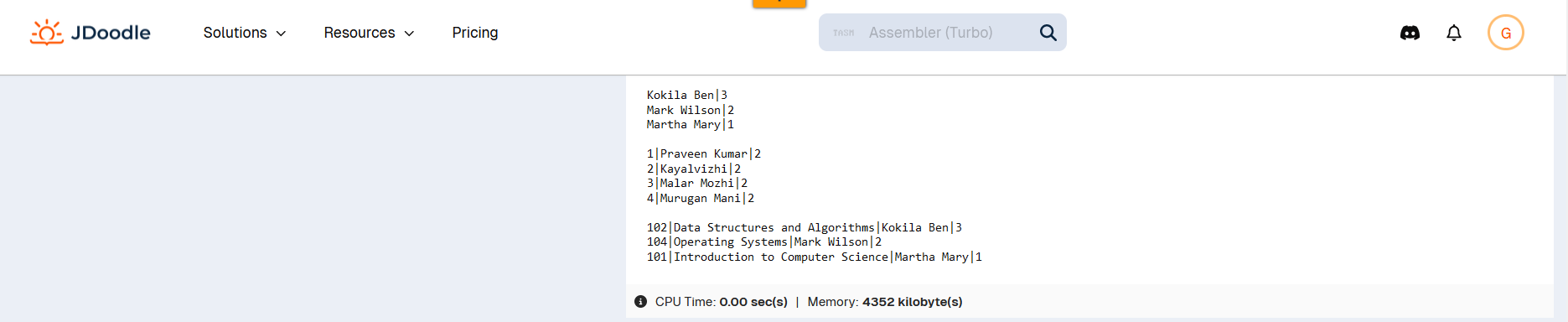
**Task 1**

<https://www.jdoodle.com/a/7Ite>





SELECT('Academic Management System - Using SQL');

SELECT(' ');

CREATE TABLE STUDENT\_INFO (

STU\_ID INT PRIMARY KEY,

STU\_NAME VARCHAR(100),

DOB DATE,

PHONE\_NO VARCHAR(15),

EMAIL\_ID VARCHAR(100),

ADDRESS VARCHAR(255)

);

INSERT INTO STUDENT\_INFO (STU\_ID, STU\_NAME, DOB, PHONE\_NO, EMAIL\_ID, ADDRESS)

VALUES

(1, 'Praveen Kumar', '2004-03-22', '0011223344', 'praveenk@example.com', '143 Gandhi St, SomeArea, INDIA'),

(2, 'Kayalvizhi', '2000-02-20', '9988776655', 'kayal@example.com', '223 T Nagar, Guindy, INDIA'),

(3, 'Malar Mozhi', '2010-04-12', '8877665544', 'mozhi@example.com', '901 Nungambakkam, Choolaimedu, INDIA'),

(4, 'Murugan Mani', '1991-08-25', '7766554433', 'mmani@example.com', '63 Local street, Besant Nagar, INDIA');

SELECT \* FROM STUDENT\_INFO;

SELECT(' ');

CREATE TABLE CoursesInfo (

COURSE\_ID INT PRIMARY KEY,

COURSE\_NAME VARCHAR(100),

COURSE\_INSTRUCTOR\_NAME VARCHAR(100)

);

INSERT INTO CoursesInfo (COURSE\_ID, COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME)

VALUES

(101, 'Introduction to Computer Science', 'Martha Mary'),

(102, 'Data Structures and Algorithms', 'Kokila Ben'),

(103, 'Database Management Systems', 'Rage Roy'),

(104, 'Operating Systems', 'Mark Wilson');

SELECT \* FROM CoursesInfo;

SELECT(' ');

CREATE TABLE EnrollmentInfo (

ENROLLMENT\_ID INT PRIMARY KEY,

STU\_ID INT,

COURSE\_ID INT,

ENROLL\_STATUS VARCHAR(20),

FOREIGN KEY (STU\_ID) REFERENCES StudentInfo (STU\_ID),

FOREIGN KEY (COURSE\_ID) REFERENCES CoursesInfo (COURSE\_ID)

);

INSERT INTO EnrollmentInfo (ENROLLMENT\_ID, STU\_ID, COURSE\_ID, ENROLL\_STATUS)

VALUES

(1, 1, 101, 'Enrolled'),

(2, 2, 102, 'Enrolled'),

(3, 3, 103, 'Not Enrolled'),

(4, 4, 104, 'Enrolled'),

(5, 1, 102, 'Enrolled'),

(6, 2, 103, 'Not Enrolled'),

(7, 3, 104, 'Enrolled'),

(8, 4, 102, 'Enrolled');

SELECT \* FROM EnrollmentInfo;

SELECT(' ');

/\*a\*/

SELECT

s.STU\_NAME,

s.PHONE\_NO,

s.EMAIL\_ID,

e.ENROLL\_STATUS

FROM

STUDENT\_INFO s

JOIN

EnrollmentInfo e ON s.STU\_ID = e.STU\_ID;

SELECT(' ');

/\*b\*/

SELECT

c.COURSE\_ID,

c.COURSE\_NAME,

c.COURSE\_INSTRUCTOR\_NAME

FROM

CoursesInfo c

JOIN

EnrollmentInfo e ON c.COURSE\_ID = e.COURSE\_ID

WHERE

e.STU\_ID = 1

AND

e.ENROLL\_STATUS = 'Enrolled';

SELECT(' ');

/\*c\*/

SELECT

COURSE\_NAME,

COURSE\_INSTRUCTOR\_NAME

FROM

CoursesInfo;

SELECT(' ');

/\*d\*/

SELECT

COURSE\_ID,

COURSE\_NAME,

COURSE\_INSTRUCTOR\_NAME

FROM

CoursesInfo

WHERE

COURSE\_ID = 103;

SELECT(' ');

/\*e\*/

SELECT

COURSE\_ID,

COURSE\_NAME,

COURSE\_INSTRUCTOR\_NAME

FROM

CoursesInfo

WHERE

COURSE\_ID IN (101, 102, 103);

SELECT(' ');

/\*Reporting - a\*/

SELECT

c.COURSE\_ID,

c.COURSE\_NAME,

COUNT(e.STU\_ID) AS NUM\_ENROLLED

FROM

CoursesInfo c

LEFT JOIN

EnrollmentInfo e ON c.COURSE\_ID = e.COURSE\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_ID, c.COURSE\_NAME;

SELECT(' ');

/\*Reporting - b\*/

SELECT

s.STU\_ID,

s.STU\_NAME,

s.PHONE\_NO,

s.EMAIL\_ID,

s.ADDRESS

FROM

STUDENT\_INFO s

JOIN

EnrollmentInfo e ON s.STU\_ID = e.STU\_ID

WHERE

e.COURSE\_ID = 101

AND

e.ENROLL\_STATUS = 'Enrolled';

SELECT(' ');

/\*Reporting - c\*/

SELECT

c.COURSE\_INSTRUCTOR\_NAME,

COUNT(DISTINCT e.STU\_ID) AS NUM\_ENROLLED\_STUDENTS

FROM

CoursesInfo c

JOIN

EnrollmentInfo e ON c.COURSE\_ID = e.COURSE\_ID

JOIN

STUDENT\_INFO s ON e.STU\_ID = s.STU\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_INSTRUCTOR\_NAME;

SELECT(' ');

/\*Reporting - d\*/

SELECT

s.STU\_ID,

s.STU\_NAME,

COUNT(e.COURSE\_ID) AS NUM\_COURSES\_ENROLLED

FROM

STUDENT\_INFO s

JOIN

EnrollmentInfo e ON s.STU\_ID = e.STU\_ID

GROUP BY

s.STU\_ID, s.STU\_NAME

HAVING

COUNT(e.COURSE\_ID) > 1;

SELECT(' ');

/\*Reporting - e\*/

SELECT

c.COURSE\_ID,

c.COURSE\_NAME,

c.COURSE\_INSTRUCTOR\_NAME,

COUNT(e.STU\_ID) AS NUM\_ENROLLED\_STUDENTS

FROM

CoursesInfo c

JOIN

EnrollmentInfo e ON c.COURSE\_ID = e.COURSE\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_ID, c.COURSE\_NAME, c.COURSE\_INSTRUCTOR\_NAME

ORDER BY

NUM\_ENROLLED\_STUDENTS DESC;

**Task 2**

1)

postgres=# CREATE DATABASE student\_database;

CREATE DATABASE

postgres=# \l

psql -U postgres -d student\_database

CREATE TABLE student\_table (

Student\_id INTEGER PRIMARY KEY,

Stu\_name TEXT NOT NULL,

Department TEXT NOT NULL,

email\_id TEXT NOT NULL,

Phone\_no NUMERIC NOT NULL,

Address TEXT,

Date\_of\_birth DATE,

Gender TEXT,

Major TEXT,

GPA NUMERIC CHECK (GPA >= 0 AND GPA <= 4.0),

Grade TEXT CHECK (Grade IN ('A', 'B', 'C', 'D', 'F'))

);

2)

psql -U postgres -d student\_database

INSERT INTO student\_table (Student\_id, Stu\_name, Department, email\_id, Phone\_no, Address, Date\_of\_birth, Gender, Major, GPA, Grade) VALUES

(1, 'John Doe', 'Computer Science', 'johndoe@example.com', 1234567890, '123 Main St', '2000-01-15', 'Male', 'Software Engineering', 3.5, 'A'),

(2, 'Jane Smith', 'Biology', 'janesmith@example.com', 2345678901, '456 Elm St', '1999-02-20', 'Female', 'Genetics', 3.7, 'A'),

(3, 'Alice Johnson', 'Mathematics', 'alicejohnson@example.com', 3456789012, '789 Maple St', '2001-03-25', 'Female', 'Statistics', 3.2, 'B'),

(4, 'Bob Brown', 'Physics', 'bobbrown@example.com', 4567890123, '101 Oak St', '2000-04-30', 'Male', 'Astrophysics', 3.8, 'A'),

(5, 'Charlie Davis', 'Chemistry', 'charliedavis@example.com', 5678901234, '202 Pine St', '1998-05-05', 'Male', 'Organic Chemistry', 2.9, 'C'),

(6, 'Dana Evans', 'History', 'danaevans@example.com', 6789012345, '303 Birch St', '1997-06-10', 'Female', 'Medieval History', 3.4, 'B'),

(7, 'Evan Foster', 'Art', 'evanfoster@example.com', 7890123456, '404 Cedar St', '1999-07-15', 'Male', 'Graphic Design', 3.6, 'A'),

(8, 'Fiona Green', 'Literature', 'fionagreen@example.com', 8901234567, '505 Spruce St', '2000-08-20', 'Female', 'Modern Literature', 3.3, 'B'),

(9, 'George Harris', 'Philosophy', 'georgeharris@example.com', 9012345678, '606 Willow St', '1998-09-25', 'Male', 'Ethics', 3.1, 'B'),

(10, 'Hannah White', 'Economics', 'hannahwhite@example.com', 1234567899, '707 Ash St', '1997-10-30', 'Female', 'Microeconomics', 3.9, 'A');

3)

psql -U postgres -d student\_database

SELECT \* FROM student\_table

ORDER BY Grade DESC;

4)

psql -U postgres -d student\_database

SELECT \* FROM student\_table

WHERE Gender = 'Male';

5)

psql -U postgres -d student\_database

SELECT \* FROM student\_table

WHERE GPA < 5.0;

6)

psql -U postgres -d student\_database

UPDATE student\_table

SET email\_id = 'new\_email@example.com', Grade = 'B'

WHERE Student\_id = 1;

7)

psql -U postgres -d student\_database

SELECT Stu\_name,

DATE\_PART('year', AGE(CURRENT\_DATE, Date\_of\_birth)) AS Age

FROM student\_table

WHERE Grade = 'B';

8)

psql -U postgres -d student\_database

SELECT Department,

Gender,

AVG(GPA) AS Average\_GPA

FROM student\_table

GROUP BY Department, Gender

ORDER BY Department, Gender;

9)

psql -U postgres -d student\_database

ALTER TABLE student\_table RENAME TO student\_info;

10)

psql -U postgres -d student\_database

SELECT Stu\_name

FROM student\_info

WHERE GPA = (

SELECT MAX(GPA)

FROM student\_info

);

**Task 3:**

1)

psql -U postgres

CREATE DATABASE EventsManagement;

\l

psql -U postgres -d EventsManagement

-- Create Events table

CREATE TABLE Events (

Event\_Id SERIAL PRIMARY KEY,

Event\_Name TEXT NOT NULL,

Event\_Date DATE NOT NULL,

Event\_Location TEXT,

Event\_Description TEXT

);

-- Create Attendees table

CREATE TABLE Attendees (

Attendee\_Id SERIAL PRIMARY KEY,

Attendee\_Name TEXT NOT NULL,

Attendee\_Phone TEXT,

Attendee\_Email TEXT,

Attendee\_City TEXT

);

-- Create Registrations table

CREATE TABLE Registrations (

Registration\_id SERIAL PRIMARY KEY,

Event\_Id INT NOT NULL,

Attendee\_Id INT NOT NULL,

Registration\_Date DATE NOT NULL,

Registration\_Amount NUMERIC,

FOREIGN KEY (Event\_Id) REFERENCES Events(Event\_Id),

FOREIGN KEY (Attendee\_Id) REFERENCES Attendees(Attendee\_Id)

);

2)

psql -U postgres -d EventsManagement

-- Insert sample data into Events table

INSERT INTO Events (Event\_Name, Event\_Date, Event\_Location, Event\_Description)

VALUES

('Conference 2024', '2024-08-15', 'New York', 'Annual conference for technology enthusiasts'),

('Workshop on Data Science', '2024-09-20', 'San Francisco', 'Hands-on workshop covering data science techniques'),

('Networking Event', '2024-07-30', 'Chicago', 'Networking event for professionals in the industry');

-- Insert sample data into Attendees table

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

VALUES

('John Doe', '123-456-7890', 'john.doe@example.com', 'New York'),

('Jane Smith', '987-654-3210', 'jane.smith@example.com', 'San Francisco'),

('Michael Johnson', '555-555-5555', 'michael.johnson@example.com', 'Chicago'),

('Emily Davis', '111-222-3333', 'emily.davis@example.com', 'Los Angeles');

-- Insert sample data into Registrations table

INSERT INTO Registrations (Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount)

VALUES

(1, 1, '2024-08-01', 100.00),

(1, 2, '2024-07-25', 100.00),

(2, 3, '2024-09-10', 75.00),

(3, 4, '2024-07-20', 50.00),

(3, 1, '2024-07-25', 50.00);

3)a)

INSERT INTO Events (Event\_Name, Event\_Date, Event\_Location, Event\_Description)

VALUES ('Tech Expo 2025', '2025-05-20', 'London', 'Annual technology exhibition showcasing innovations');

3)b)

UPDATE Events

SET Event\_Name = 'Updated Event Name',

Event\_Date = '2025-06-15',

Event\_Location = 'Paris',

Event\_Description = 'Updated description for the event'

WHERE Event\_Id = 1; -- Replace with the actual Event\_Id of the event you want to update

3)c)

DELETE FROM Events

WHERE Event\_Id = 1; -- Replace with the actual Event\_Id of the event you want to delete

4)a)

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

VALUES ('Alice Johnson', '555-123-4567', 'alice.johnson@example.com', 'New York');

4)b)

INSERT INTO Registrations (Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount)

VALUES (1, 1, '2024-07-03', 0.00);

4)c)

SELECT Event\_Id, Event\_Name, Event\_Date, Event\_Location, Event\_Description

FROM Events;

SELECT A.Attendee\_Id, A.Attendee\_Name, A.Attendee\_Email, A.Attendee\_City

FROM Attendees A

JOIN Registrations R ON A.Attendee\_Id = R.Attendee\_Id

WHERE R.Event\_Id = 1; -- Replace with the Event\_Id of the event you are interested in

SELECT E.Event\_Id, E.Event\_Name, COUNT(R.Attendee\_Id) AS Attendee\_Count

FROM Events E

LEFT JOIN Registrations R ON E.Event\_Id = R.Event\_Id

GROUP BY E.Event\_Id, E.Event\_Name

ORDER BY Attendee\_Count DESC;

**Task 4:**

1)

CREATE DATABASE sales\_database;

\l

CREATE TABLE sales\_sample (

Product\_Id INTEGER,

Region VARCHAR(50),

Date DATE,

Sales\_Amount NUMERIC

);

2)

INSERT INTO sales\_sample (Product\_Id, Region, Date, Sales\_Amount)

VALUES

(1, 'East', '2024-01-01', 1000.00),

(2, 'West', '2024-01-02', 1500.50),

(3, 'North', '2024-01-03', 800.75),

(1, 'East', '2024-01-04', 1200.25),

(2, 'West', '2024-01-05', 1350.30),

(3, 'North', '2024-01-06', 950.00),

(1, 'East', '2024-01-07', 1100.50),

(2, 'West', '2024-01-08', 1400.75),

(3, 'North', '2024-01-09', 850.25),

(1, 'East', '2024-01-10', 1300.20);

3)a)

SELECT

Region,

Product\_Id,

SUM(Sales\_Amount) AS Total\_Sales\_Amount

FROM

sales\_sample

GROUP BY

Region, Product\_Id

ORDER BY

Region, Product\_Id;

3)b)

SELECT

Product\_Id,

Region,

SUM(Sales\_Amount) AS Total\_Sales\_Amount

FROM

sales\_sample

GROUP BY

Product\_Id, Region

ORDER BY

Product\_Id, Region;

SELECT

COALESCE(Product\_Id::TEXT, 'Total') AS Product\_Id,

COALESCE(Region, 'Total') AS Region,

COALESCE(CAST(Date AS TEXT), 'Total') AS Date,

SUM(Sales\_Amount) AS Total\_Sales\_Amount

FROM

sales\_sample

GROUP BY

CUBE(Product\_Id, Region, Date)

ORDER BY

Product\_Id, Region, Date;

3)c)

SELECT

Product\_Id,

Region,

Date,

Sales\_Amount

FROM

sales\_sample

WHERE

Region = 'East';

3)d)

SELECT

Product\_Id,

Region,

Date,

Sales\_Amount

FROM

sales\_sample

WHERE

Date BETWEEN '2024-01-01' AND '2024-01-10';

3)e)

SELECT

Product\_Id,

Region,

Date,

Sales\_Amount

FROM

sales\_sample

WHERE

Product\_Id = 1

AND Region = 'East'

AND Date BETWEEN '2024-01-01' AND '2024-01-10';