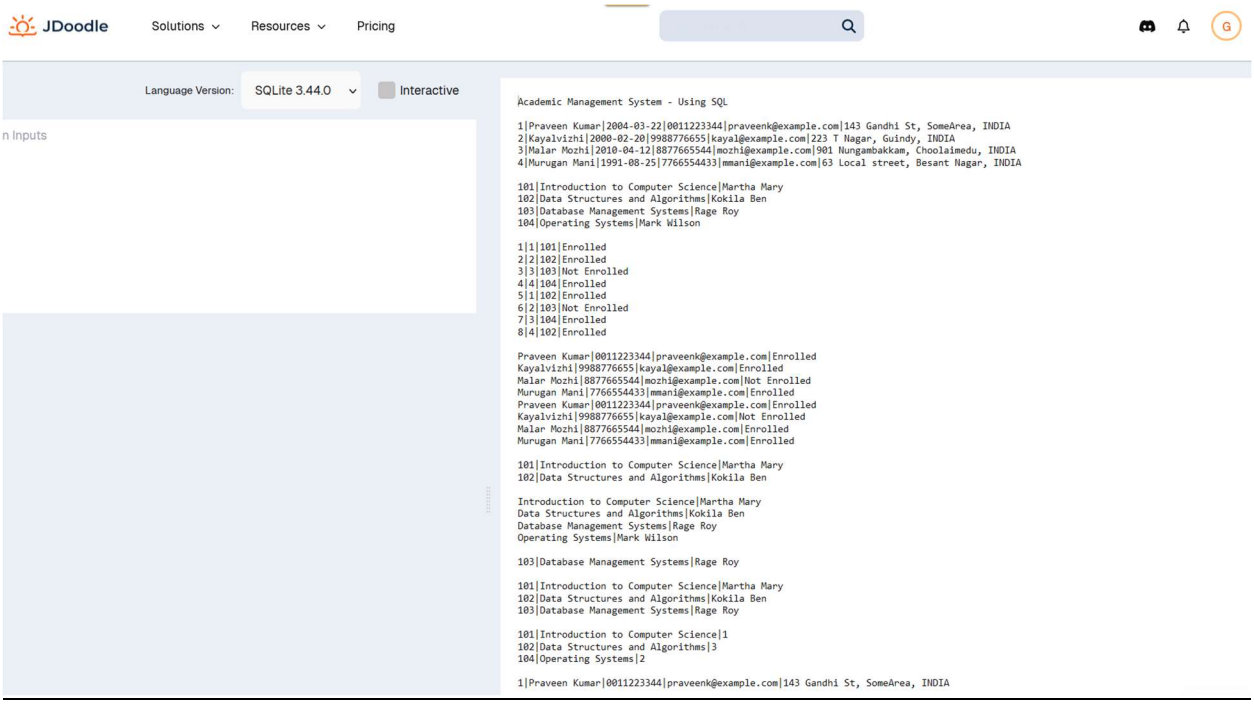
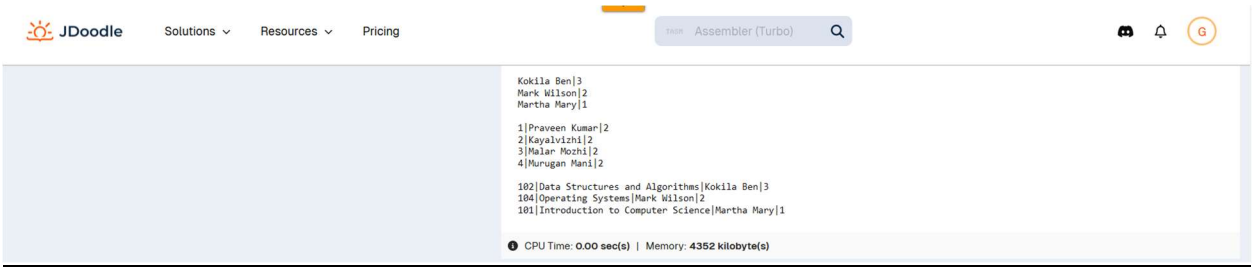
Task 1

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';
```