DB Assignment

- 1. Create the tables below in the database. Use foreign keys and primary keys as required.
 - a. Create a table called as student with the following columns student_id, first_name, last_name ,birthdate , department_id ,address_id .

```
create table student(
    student_id int primary key,
    first_name varchar(100),
    last_name varchar(100),
    birthdate date,
    department_id int,
    address_id int,
    foreign key(department_id) references department(department_id),
    foreign key(address_id) references address(address_id));
```

b. Create Address table with following columns address_id , street_address, city, State, postal_code

```
create table Address(
   address_id int primary key,
   street_address varchar(255),
   city varchar(100),
   state varchar(100),
   postal_code varchar(20));
```

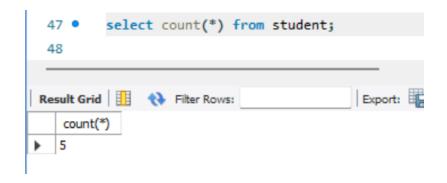
c. Create department table department_id, department name. Make sure you are using the right data type against all the columns.

```
create table Department(
  department_id int primary key,
  department_name varchar(100));
```

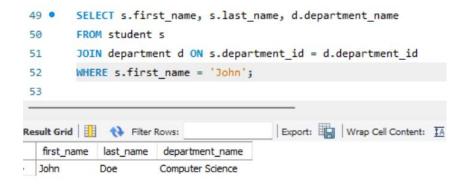
2. Use Sample data from <u>sampledata.txt</u> to insert data into the database

```
load data infile 'C:/Java/TrainingAtBounteous/28-05-2025_Day5/sampledata.txt'
into table student
fields terminated by 'n' lines terminated by '\n';
```

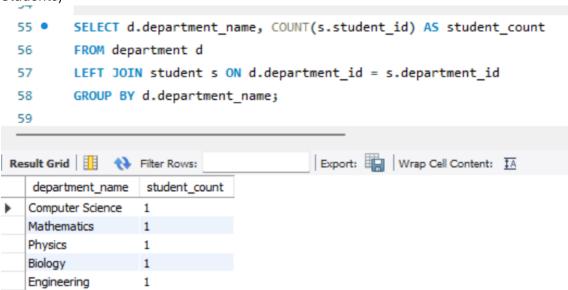
3. Write a query to find the total number of students.



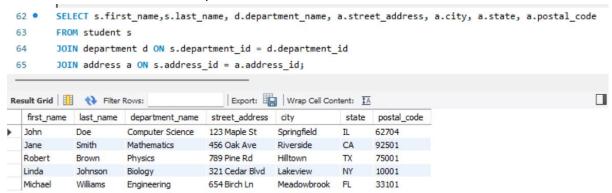
4. Write a query to find which department john belongs to.



5. List All Departments with Their Number of Students (Including Departments with No Students)



6. Select all students with their department and address.



7. Find all students who are in the 'Computer Science' department

```
68
        SELECT s.*
69
        FROM student s
        JOIN department d ON s.department_id = d.department_id
 70
        WHERE d.department_name = 'Computer Science';
71
                                        Export: Wrap Cell Content: IA
student_id | first_name
                                          department_id
                      last_name
                               birthdate
                                                      address_id
            John
                     Doe
                               2000-05-15
                                                      1001
```

8. Update Jane's city name to New York.

```
update address set city='New York'
where address_id=(select address_id from student where first_name='Jane');
```

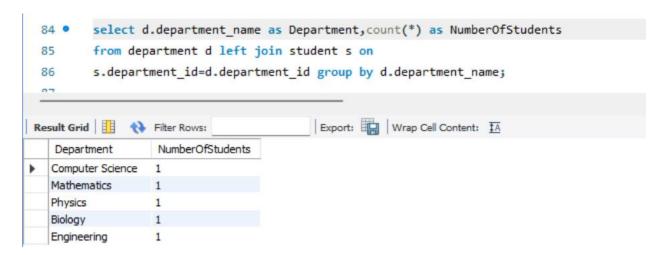
9. Delete a student from the student table.

```
DELETE FROM student
WHERE student_id = 5;
```

10. Select all students with their department and address in New York.

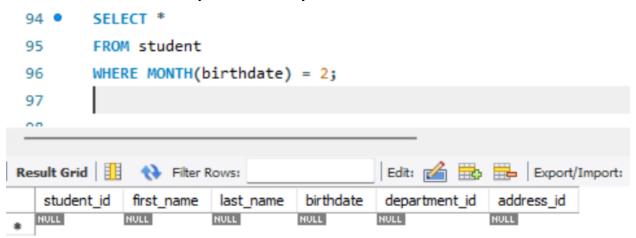
```
78 •
        SELECT s.*, d.department_name, a.city
 79
        FROM student s
        JOIN department d ON s.department_id = d.department_id
 80
        JOIN address a ON s.address_id = a.address_id
        WHERE a.city = 'New York';
 82
                                         Export: Wrap Cell Content: IA
student id
                                           department_id address_id
                                                                  department_name
            first name
                      last name
                                birthdate
                                                                                 city
 2
                      Smith
                                1999-10-20
                                          102
                                                       1002
                                                                 Mathematics
                                                                                New York
            Jane
```

11. Count how many students are in each department

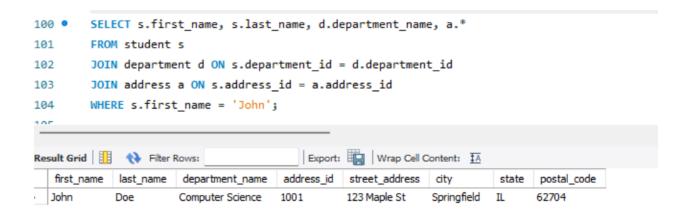


```
89 •
         SELECT s.*
 90
         FROM student s
         JOIN address a ON s.address id = a.address id
 91
         WHERE a.city = 'Springfield';
 92
                                              Export: Wrap Cell Content: $\frac{1}{4}
Result Grid
               Filter Rows:
   student_id
                                    birthdate
                                                               address_id
              first_name
                         last_name
                                                 department_id
              John
                         Doe
                                    2000-05-15
                                                101
                                                               1001
```

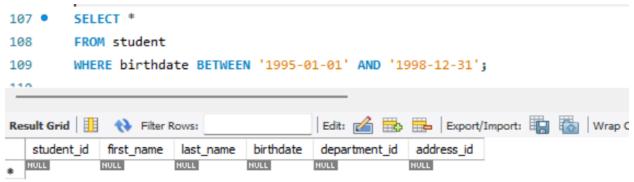
13. Select students whose birthday falls in February



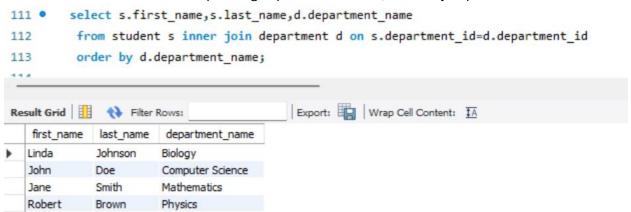
14. Get the department and address details for a specific student, example john



15. Find all students who are born within 1995 to 1998



16. List all students and their corresponding department names, sorted by department



17. Find the number of students in each department who are living in 'Champaign'

18. Retrieve the names of students who live on 'Pine' street

```
SELECT s.first_name, s.last_name

FROM student s

JOIN address a ON s.address_id = a.address_id

WHERE a.street_address = 'Pine';

Result Grid Filter Rows:

Filter Rows:

Export: Wrap Cell Cont
```

19. Update the department of a student with student_id = 6 to 'Mechanical Engineering'

```
update student set department_id=(select department_id from department
where department_name='Mechanical Engineering') where student_id=6;
```

20. Find the student(s) who live in the city 'Chicago' and are in the 'Mathematics' department

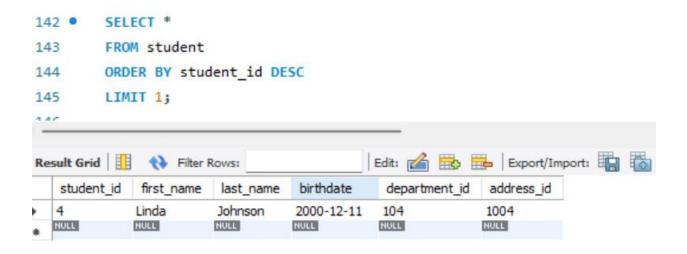
21. List all students who have an address in 'Urbana' or 'Peoria'

```
FROM student s

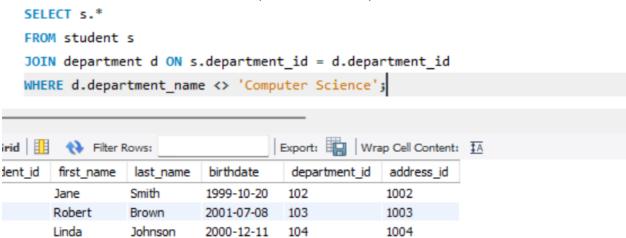
JOIN address a ON s.address_id = a.address_id

WHERE a.city IN ('Urbana', 'Peoria');
```

22. Find the student with the highest student_id



23. Find all students who are not in the 'Computer Science' department



24. Count the total number of addresses in the 'Champaign' city

```
SELECT COUNT(*) AS address_count

FROM address

WHERE city = 'Champaign';
```

25. Find the name of the student who lives at '520 Pine Rd'

```
SELECT s.first_name, s.last_name

FROM student s

JOIN address a ON s.address_id = a.address_id

WHERE a.street_address = '520 Pine Rd';
```

26. Get the average age of students in the 'Electrical Engineering' department

```
SELECT AVG(YEAR(CURDATE()) - YEAR(birthdate)) AS average_age
FROM student s

JOIN department d ON s.department_id = d.department_id

WHERE d.department_name = 'Electrical Engineering';
```

27. List the students, their department, and the city where they live, but only for those in departments starting with 'M'

```
SELECT s.first_name, s.last_name, d.department_name, a.city
167 •
168
        FROM student s
169
        JOIN department d ON s.department id = d.department id
        JOIN address a ON s.address id = a.address id
170
        WHERE d.department name LIKE 'M%';
171
Export: Wrap Cell Content: ‡A
   first name
            last name
                     department name
                                    city
  Jane
           Smith
                    Mathematics
                                   New York
```

28. Delete a student from the 'Mechanical Engineering' department

```
DELETE FROM student

HHERE department_id = (
    SELECT department_id
    FROM department

WHERE department_name = 'Mechanical Engineering'
    LIMIT 1)

LIMIT 1;
```

Open PG Admin and open query tool and select any database of your choice.

Click on "Open file" and select order.sql from your device and execute it.

Questions:

1. Retrieve All Orders with Their Customer Details and Current Status

```
SELECT o.order_id, o.order_date,
298
               c.customer_id, c.first_name, c.last_name, c.email, c.phone_number,
299
               s.status name
300
        FROM order schema.orders o
        JOIN order_schema.customer c ON o.customer_id = c.customer_id
301
        JOIN order_schema.status s ON o.status_id = s.status_id
302
        ORDER BY o.order_date DESC;
303
304
Export: Wrap Cell Content: IA
  order_id order_date customer_id first_name last_name
                                                                         phone_number status_name
                                                    email
                                                   john.doe@example.com
          2025-02-18 1
                                John
                                                                         555-1234
                                                                                     Cancelled
  3
          2025-02-17 3
                                          Jones
                                                  emily.jones@example.com
                                                                        555-8765
                                                                                     Shipped
          2025-02-16 2
                                          Smith
                                                   jane.smith@example.com
                                                                         555-5678
                                                                                      Pending
                                Jane
  1
          2025-02-15 1
                                John
                                          Doe
                                                   john.doe@example.com
                                                                        555-1234
                                                                                     Shipped
```

2. Get the Total Value of Orders for a Given Customer in a Specific Time Period

```
SELECT o.customer_id, SUM(oi.quantity * oi.price) AS total_order_value

FROM order_schema.orders o

JOIN order_schema.order_items oi ON o.order_id = oi.order_id

WHERE o.customer_id = 4

AND o.order_date BETWEEN 2022-05-12 AND 2025-03-01

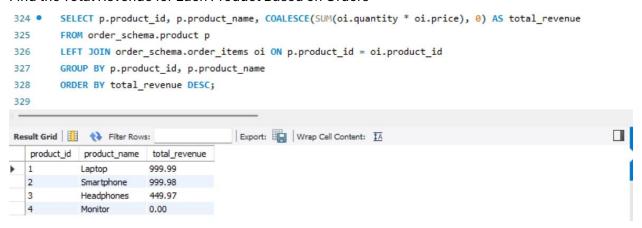
GROUP BY o.customer_id;
```



3. Find the Most Expensive Order by Customer

```
SELECT o.customer_id, o.order_id, SUM(oi.quantity * oi.price) AS order_total
315 •
        FROM order schema.orders o
316
        JOIN order_schema.order_items oi ON o.order_id = oi.order_id
317
        GROUP BY o.customer_id, o.order_id
318
        ORDER BY order_total DESC
319
320
        LIMIT 1;
321
                                                                               1
Export: Wrap Cell Content: TA Fetch rows:
   customer_id
             order_id order_total
                     1499.98
1
             1
```

4. Find the Total Revenue for Each Product Based on Orders



5. Write a query to retrieve the order ID, customer ID, and the total amount of each order. If the total amount is null, display '0.00' instead.

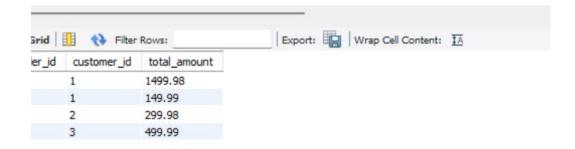
```
SELECT o.order_id, o.customer_id,

COALESCE(SUM(oi.quantity * oi.price), 0.00) AS total_amount

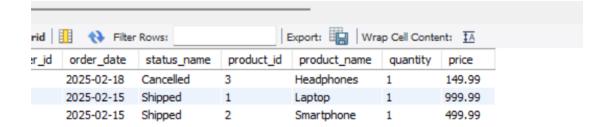
FROM order_schema.orders o

LEFT JOIN order_schema.order_items oi ON o.order_id = oi.order_id

GROUP BY o.order_id, o.customer_id;
```



6. Retrieve the Order History of a Specific Customer Along with Product Details



7. Get the Average Order Value Per Customer in the Last 30 Days.

```
SELECT o.customer_id, AVG(order_total) AS avg_order_value

FROM (

SELECT o.order_id, o.customer_id, SUM(oi.quantity * oi.price) AS order_total

FROM order_schema.orders o

JOIN order_schema.order_items oi ON o.order_id = oi.order_id

WHERE o.order_date >= CURRENT_DATE - INTERVAL '30 days'

GROUP BY o.order_id, o.customer_id

) AS order_totals

GROUP BY o.customer_id;
```

8. Get the Top 5 Products with the Highest Number of Orders.

9. Get the Customers Who Have Not Placed Any Orders in the Last 60 Days

```
SELECT c.customer_id, c.first_name, c.last_name, c.email
FROM order_schema.customer c
LEFT JOIN order_schema.orders o ON c.customer_id = o.customer_id
AND o.order_date >= CURRENT_DATE - INTERVAL '60 days'
WHERE o.order_id IS NULL;
```

10. List the Orders with Products Ordered More Than Once, Sorted by Order Date

```
SELECT o.order_id, o.order_date, p.product_name, oi.quantity
378
379
        FROM order schema.orders o
380
        JOIN order_schema.order_items oi ON o.order_id = oi.order_id
381
        JOIN order schema.product p ON oi.product id = p.product id
382
        WHERE oi.quantity > 1
        ORDER BY o.order_date DESC;
383
384
385
                                       Export: Wrap Cell Content: TA
order_id order_date
                     product_name
                                 quantity
          2025-02-16
                    Headphones
```

11. Retrieve the Number of Orders and Total Revenue for Each Status

```
387 •
        SELECT s.status name,
                COUNT(DISTINCT o.order_id) AS order_count,
388
389
                COALESCE(SUM(oi.quantity * oi.price), 0) AS total_revenue
         FROM order_schema.status s
390
        LEFT JOIN order_schema.orders o ON s.status_id = o.status_id
391
         LEFT JOIN order_schema.order_items oi ON o.order_id = oi.order_id
392
        GROUP BY s.status name;
393
394
395
                                          Export: Wrap Cell Content: TA
Result Grid
             Filter Rows:
   status_name order_count total_revenue
  Cancelled
                          149.99
  Pending
                          299.98
              1
  Shipped
              2
                          1999.97
```

12. Customers Who Have Ordered More Than a Specific Product (e.g., "Laptop")

```
SELECT c.customer_id, c.first_name, c.last_name, SUM(oi.quantity) AS total_quantity

FROM order_schema.customer c

JOIN order_schema.orders o ON c.customer_id = o.customer_id

JOIN order_schema.order_items oi ON o.order_id = oi.order_id

JOIN order_schema.product p ON oi.product_id = p.product_id

WHERE p.product_name = 'Laptop'

GROUP BY c.customer_id, c.first_name, c.last_name

HAVING SUM(oi.quantity) > 1;

Grid 
Filter Rows:

Export: Wrap Cell Content: A

Stomer_id first_name last_name total_quantity
```

13. Find the Products That Have Never Been Ordered

14. Get the Total Quantity of Products Ordered in the Last 7 Days

```
SELECT COALESCE(SUM(oi.quantity), 0) AS total_quantity_ordered

FROM order_schema.order_items oi

JOIN order_schema.orders o ON oi.order_id = o.order_id

WHERE o.order_date >= CURRENT_DATE - INTERVAL '7 days';
```

15. Create a view named product_details that includes all columns from the product table.

```
CREATE VIEW order_schema.product_details AS

SELECT *

FROM order_schema.product;
```

16. Create a view named order_summary that includes the order_id, customer_id, order_date, total_amount, and status_name (from the status table) for each order.