1. Create the tables below in the database. Use foreign keys and primary keys as required.
   1. Create a table called as student with the following columns student\_id, first\_name, last\_name ,birthdate , department\_id ,address\_id .
   2. Create Address table with following columns address\_id , street\_address, city, State, postal\_code
   3. 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) NOT NULL

);

CREATE TABLE address(

address\_id INT PRIMARY KEY,

street\_address VARCHAR(150) NOT NULL,

city VARCHAR(100) NOT NULL,

state CHAR(2) NOT NULL,

postal\_code VARCHAR(10) NOT NULL

);

CREATE TABLE student(

student\_id INT PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

birthdate DATE NOT NULL,

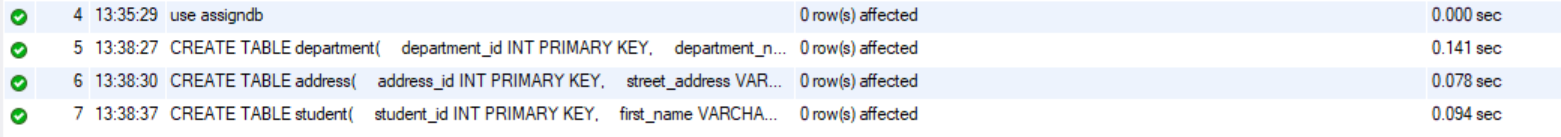
department\_id INT,

address\_id INT,

FOREIGN KEY (department\_id) REFERENCES department(department\_id),

FOREIGN KEY (address\_id) REFERENCES address(address\_id)

);



1. Use Sample data from [sampledata.txt](https://hs2solutions-my.sharepoint.com/:t:/g/personal/iqbal_sarkar_bounteous_com/ET8VdCEdjTJNoaWCrFPFxfwBq9bpLp9HkT7ycwnltym8qA?e=xTKw8M) to insert data into the database

INSERT INTO department VALUES

(1, 'Computer Science'),

(2, 'Mechanical Engineering'),

(3, 'Electrical Engineering'),

(4, 'Civil Engineering'),

(5, 'Mathematics'),

(6, 'Biology');

INSERT INTO address VALUES

(1, '123 Elm St', 'Springfield', 'IL', '62701'),

(2, '456 Oak St', 'Decatur', 'IL', '62521'),

(3, '789 Pine St', 'Champaign', 'IL', '61820'),

(4, '102 Birch Rd', 'Peoria', 'IL', '61602'),

(5, '205 Cedar Ave', 'Chicago', 'IL', '60601'),

(6, '310 Maple Dr', 'Urbana', 'IL', '61801'),

(7, '415 Oak Blvd', 'Champaign', 'IL', '61821'),

(8, '520 Pine Rd', 'Carbondale', 'IL', '62901');

INSERT INTO student VALUES

(1, 'John', 'Doe', '1995-04-15', 1, 1),

(2, 'Jane', 'Smith', '1996-07-22', 2, 2),

(3, 'Alice', 'Johnson', '1994-11-30', 3, 3),

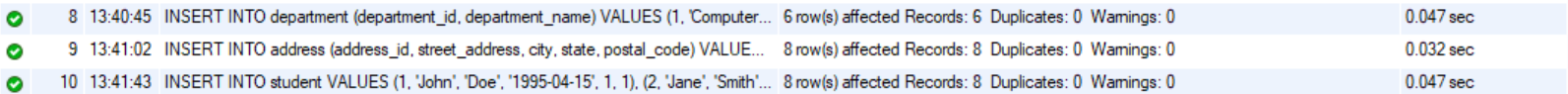
(4, 'Michael', 'Brown', '1997-02-19', 4, 4),

(5, 'Sophia', 'Davis', '1998-01-05', 5, 5),

(6, 'Daniel', 'Wilson', '1995-06-10', 6, 6),

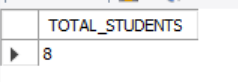
(7, 'Olivia', 'Martinez', '1997-11-25', 1, 7),

(8, 'Ethan', 'Miller', '1996-03-30', 2, 8);



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

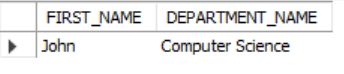
SELECT COUNT(\*) AS TOTAL\_STUDENTS FROM STUDENT;



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

SELECT FIRST\_NAME, DEPARTMENT\_NAME FROM STUDENT INNER JOIN DEPARTMENT

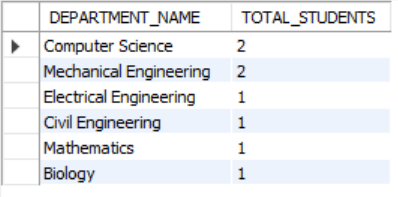
ON STUDENT.DEPARTMENT\_ID = DEPARTMENT.DEPARTMENT\_ID WHERE FIRST\_NAME = "John";



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

SELECT D.DEPARTMENT\_NAME, COUNT(S.STUDENT\_ID) AS TOTAL\_STUDENTS

FROM DEPARTMENT D LEFT JOIN STUDENT S

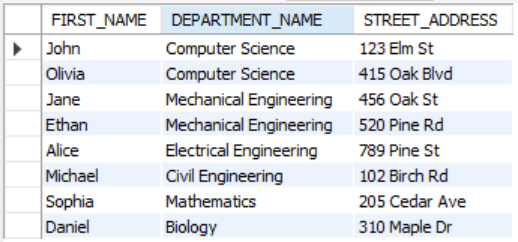
ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID GROUP BY D.DEPARTMENT\_NAME;

1. Select all students with their department and address.

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME, A.STREET\_ADDRESS

FROM DEPARTMENT D INNER JOIN STUDENT S INNER JOIN ADDRESS A

ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID AND S.ADDRESS\_ID = A.ADDRESS\_ID;

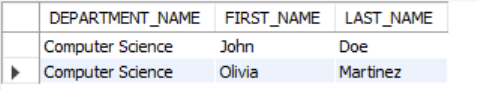


1. Find all students who are in the 'Computer Science' department

SELECT DEPARTMENT\_NAME, FIRST\_NAME, LAST\_NAME FROM STUDENT INNER JOIN DEPARTMENT

ON STUDENT.DEPARTMENT\_ID = DEPARTMENT.DEPARTMENT\_ID

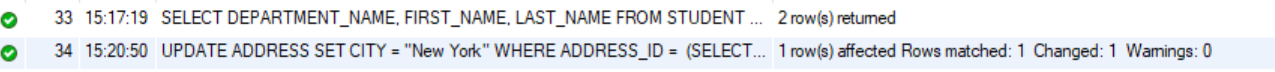
WHERE DEPARTMENT\_NAME = "Computer Science";

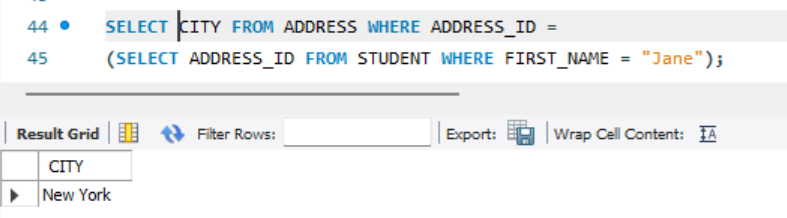


1. Update Jane’s city name to New York.

UPDATE ADDRESS SET CITY = "New York" WHERE ADDRESS\_ID =

(SELECT ADDRESS\_ID FROM STUDENT WHERE FIRST\_NAME = "Jane");





1. Delete a student from the student table.

DELETE FROM STUDENT WHERE STUDENT\_ID = 8;



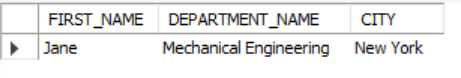
1. Select all students with their department and address in New York.

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME, A.CITY

FROM DEPARTMENT D INNER JOIN STUDENT S INNER JOIN ADDRESS A

ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID AND S.ADDRESS\_ID = A.ADDRESS\_ID

WHERE A.CITY = "New York";

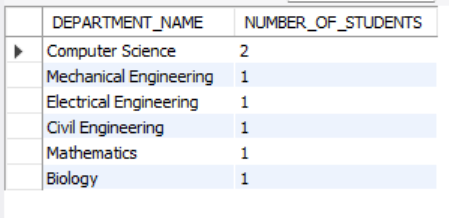


1. Count how many students are in each department

SELECT D.DEPARTMENT\_NAME, COUNT(S.STUDENT\_ID) AS NUMBER\_OF\_STUDENTS

FROM DEPARTMENT D LEFT JOIN STUDENT S ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID

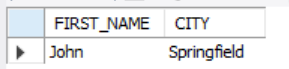
GROUP BY D.DEPARTMENT\_NAME;



1. Find students who live in 'Springfield'

SELECT S.FIRST\_NAME, A.CITY FROM STUDENT S INNER JOIN ADDRESS A

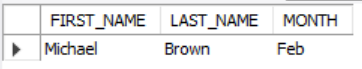
ON S.ADDRESS\_ID = A.ADDRESS\_ID WHERE A.CITY = "Springfield";



1. Select students whose birthday falls in February

SELECT FIRST\_NAME, LAST\_NAME, DATE\_FORMAT(BIRTHDATE, '%b') as MONTH

FROM STUDENT WHERE MONTH(BIRTHDATE) = 2;

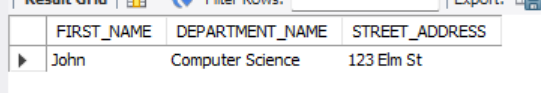


1. Get the department and address details for a specific student, example john

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME, A.STREET\_ADDRESS

FROM DEPARTMENT D INNER JOIN STUDENT S INNER JOIN ADDRESS A

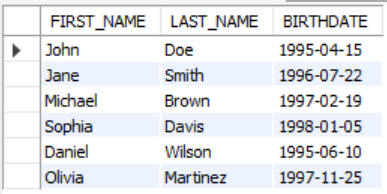
ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID AND S.ADDRESS\_ID = A.ADDRESS\_ID

WHERE S.FIRST\_NAME = "John"; 

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

SELECT FIRST\_NAME, LAST\_NAME, BIRTHDATE FROM STUDENT

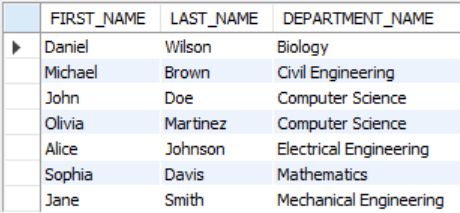
WHERE YEAR(BIRTHDATE)>=1995 AND YEAR(BIRTHDATE)<=1998;



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

SELECT FIRST\_NAME, LAST\_NAME, DEPARTMENT\_NAME FROM STUDENT INNER JOIN DEPARTMENT

ON STUDENT.DEPARTMENT\_ID = DEPARTMENT.DEPARTMENT\_ID ORDER BY DEPARTMENT\_NAME;



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

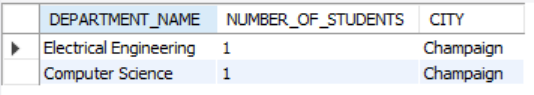
SELECT DEPARTMENT\_NAME, NUMBER\_OF\_STUDENTS, CITY FROM

(SELECT S.ADDRESS\_ID, D.DEPARTMENT\_NAME, COUNT(S.STUDENT\_ID) AS NUMBER\_OF\_STUDENTS

FROM DEPARTMENT D LEFT JOIN STUDENT S ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID

GROUP BY D.DEPARTMENT\_NAME, S.ADDRESS\_ID) AS T1 INNER JOIN ADDRESS A

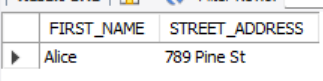
ON T1.ADDRESS\_ID = A.ADDRESS\_ID WHERE CITY = "Champaign";



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

SELECT S.FIRST\_NAME, A.STREET\_ADDRESS FROM STUDENT S INNER JOIN ADDRESS A

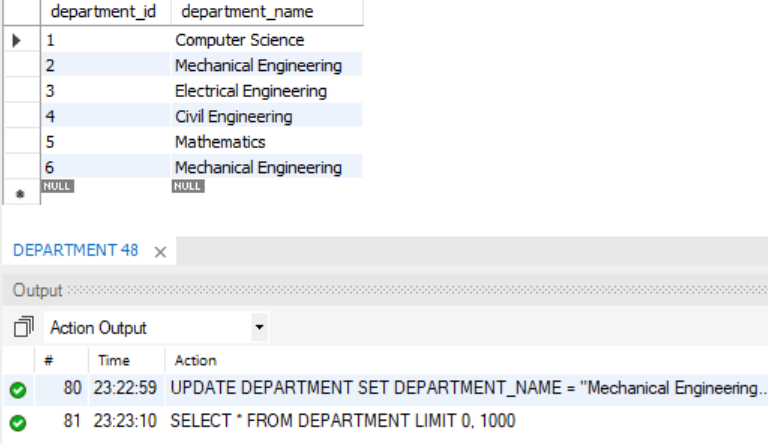
ON S.ADDRESS\_ID = A.ADDRESS\_ID WHERE A.STREET\_ADDRESS LIKE "%pine%";



1. Update the department of a student with student\_id = 6 to 'Mechanical Engineering'

UPDATE DEPARTMENT SET DEPARTMENT\_NAME = "Mechanical Engineering" WHERE DEPARTMENT\_ID =

(SELECT DEPARTMENT\_ID FROM STUDENT WHERE STUDENT\_ID = 6);



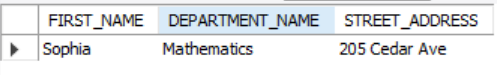
1. Find the student(s) who live in the city 'Chicago' and are in the 'Mathematics' department

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME, A.STREET\_ADDRESS

FROM DEPARTMENT D INNER JOIN STUDENT S INNER JOIN ADDRESS A

ON D.DEPARTMENT\_ID = S.DEPARTMENT\_ID AND S.ADDRESS\_ID = A.ADDRESS\_ID

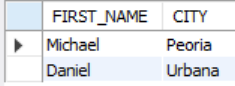
WHERE D.DEPARTMENT\_NAME = "Mathematics" AND A.CITY = "Chicago";



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

SELECT S.FIRST\_NAME, A.CITY FROM STUDENT S INNER JOIN ADDRESS A

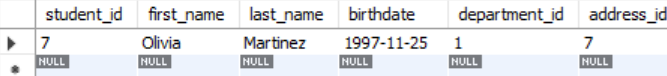
ON S.ADDRESS\_ID = A.ADDRESS\_ID WHERE A.CITY = "Urbana" OR A.CITY = "Peoria";



1. Find the student with the highest student\_id

SELECT \* FROM STUDENT

WHERE STUDENT\_ID = (SELECT MAX(STUDENT\_ID) FROM STUDENT);

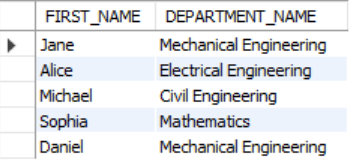


1. Find all students who are not in the 'Computer Science' department

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME FROM STUDENT S INNER JOIN DEPARTMENT D

ON S.DEPARTMENT\_ID = D.DEPARTMENT\_ID

WHERE D.DEPARTMENT\_NAME <> 'Computer Science';



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

SELECT COUNT(\*) AS TOTAL\_ADDRESSES

FROM ADDRESS

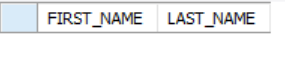
WHERE CITY = 'Champaign';



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

SELECT S.FIRST\_NAME, S.LAST\_NAME FROM STUDENT S INNER JOIN ADDRESS A

ON S.ADDRESS\_ID = A.ADDRESS\_ID WHERE A.STREET\_ADDRESS LIKE "520 Pine Rd";

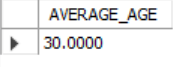
(STUDENT RECORD WAS DELETED PREV)

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

SELECT AVG(TIMESTAMPDIFF(YEAR, BIRTHDATE, CURDATE())) AS AVERAGE\_AGE

FROM STUDENT S INNER JOIN DEPARTMENT D ON S.DEPARTMENT\_ID = D.DEPARTMENT\_ID

WHERE D.DEPARTMENT\_NAME = 'Electrical Engineering';

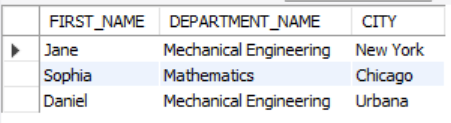


1. List the students, their department, and the city where they live, but only for those in departments starting with 'M'

SELECT S.FIRST\_NAME, D.DEPARTMENT\_NAME, A.CITY FROM STUDENT S INNER JOIN DEPARTMENT D

ON S.DEPARTMENT\_ID = D.DEPARTMENT\_ID INNER JOIN ADDRESS A

ON S.ADDRESS\_ID = A.ADDRESS\_ID WHERE D.DEPARTMENT\_NAME LIKE 'M%';



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

DELETE FROM STUDENT

WHERE DEPARTMENT\_ID = (SELECT DEPARTMENT\_ID FROM DEPARTMENT WHERE DEPARTMENT\_NAME = 'Mechanical Engineering' LIMIT 1);



Download [order.sql](https://hs2solutions-my.sharepoint.com/:u:/g/personal/iqbal_sarkar_bounteous_com/ERszToxU3wJNo1HXV0zhiNgBMx8PQdMY207902wVY8Mnhg?e=qqdFSF)

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

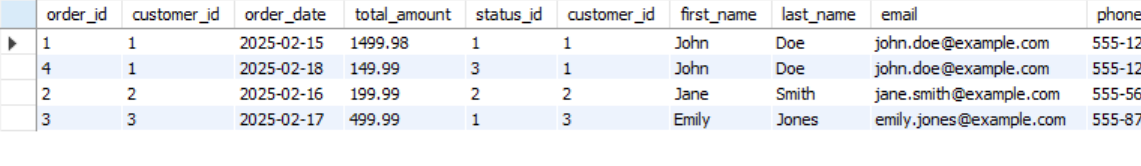
Click on “Open file” and select [order.sql](https://hs2solutions-my.sharepoint.com/:u:/g/personal/iqbal_sarkar_bounteous_com/ERszToxU3wJNo1HXV0zhiNgBMx8PQdMY207902wVY8Mnhg?e=qqdFSF) from your device and execute it.

Questions:

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

SELECT \* FROM orders o JOIN customer c

ON o.customer\_id = c.customer\_id JOIN status s ON o.status\_id = s.status\_id;



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

SELECT SUM(TOTAL\_AMOUNT) AS TOTAL\_ORDER\_VALUE FROM ORDERS

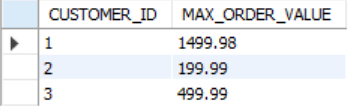
WHERE CUSTOMER\_ID = 1 AND ORDER\_DATE BETWEEN '2025-02-01' AND '2025-02-28';



1. Find the Most Expensive Order by Customer

SELECT CUSTOMER\_ID, MAX(TOTAL\_AMOUNT) AS MAX\_ORDER\_VALUE

FROM ORDERS GROUP BY CUSTOMER\_ID;

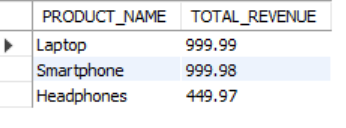


1. Find the Total Revenue for Each Product Based on Orders

SELECT P.PRODUCT\_NAME, SUM(OI.QUANTITY \* OI.PRICE) AS TOTAL\_REVENUE

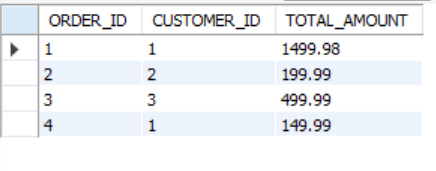
FROM ORDER\_ITEMS OI JOIN PRODUCT P ON OI.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY P.PRODUCT\_NAME;



1. 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 ORDER\_ID, CUSTOMER\_ID, COALESCE(TOTAL\_AMOUNT, 0.00) AS TOTAL\_AMOUNT FROM ORDERS;



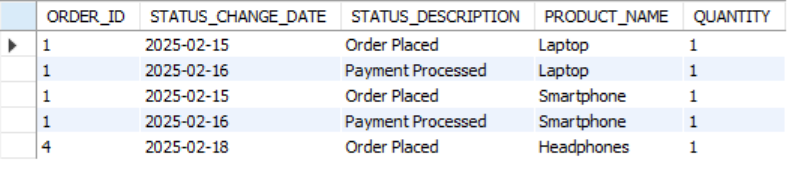
1. Retrieve the Order History of a Specific Customer Along with Product Details

SELECT OH.ORDER\_ID, OH.STATUS\_CHANGE\_DATE, OH.STATUS\_DESCRIPTION, P.PRODUCT\_NAME, OI.QUANTITY

FROM ORDER\_HISTORY OH JOIN ORDERS O ON OH.ORDER\_ID = O.ORDER\_ID

JOIN ORDER\_ITEMS OI ON O.ORDER\_ID = OI.ORDER\_ID JOIN PRODUCT P ON OI.PRODUCT\_ID = P.PRODUCT\_ID

WHERE O.CUSTOMER\_ID = 1;

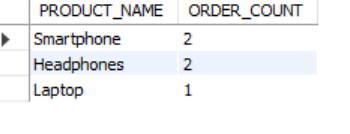


1. Get the Average Order Value Per Customer in the Last 30 Days.
2. Get the Top 5 Products with the Highest Number of Orders.

SELECT P.PRODUCT\_NAME, COUNT(OI.ORDER\_ID) AS ORDER\_COUNT

FROM ORDER\_ITEMS OI JOIN PRODUCT P ON OI.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY P.PRODUCT\_NAME ORDER BY ORDER\_COUNT DESC LIMIT 5;



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

SELECT C.CUSTOMER\_ID, C.FIRST\_NAME, C.LAST\_NAME

FROM CUSTOMER C

LEFT JOIN ORDERS O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

GROUP BY C.CUSTOMER\_ID

HAVING MAX(O.ORDER\_DATE) IS NULL OR MAX(O.ORDER\_DATE) < MAX(O.ORDER\_DATE) - 60;

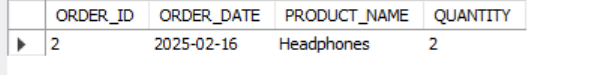
1. List the Orders with Products Ordered More Than Once, Sorted by Order Date

SELECT O.ORDER\_ID, O.ORDER\_DATE, P.PRODUCT\_NAME, OI.QUANTITY

FROM ORDERS O JOIN ORDER\_ITEMS OI ON O.ORDER\_ID = OI.ORDER\_ID

JOIN PRODUCT P ON OI.PRODUCT\_ID = P.PRODUCT\_ID

WHERE OI.QUANTITY > 1 ORDER BY O.ORDER\_DATE;



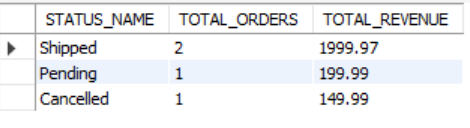
1. Retrieve the Number of Orders and Total Revenue for Each Status

SELECT S.STATUS\_NAME, COUNT(O.ORDER\_ID) AS TOTAL\_ORDERS,

COALESCE(SUM(O.TOTAL\_AMOUNT), 0) AS TOTAL\_REVENUE

FROM STATUS S LEFT JOIN ORDERS O ON S.STATUS\_ID = O.STATUS\_ID

GROUP BY S.STATUS\_NAME;



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

SELECT DISTINCT C.CUSTOMER\_ID, C.FIRST\_NAME, C.LAST\_NAME FROM CUSTOMER C

JOIN ORDERS O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID JOIN ORDER\_ITEMS OI ON O.ORDER\_ID = OI.ORDER\_ID

JOIN PRODUCT P ON OI.PRODUCT\_ID = P.PRODUCT\_ID WHERE P.PRODUCT\_NAME = 'Laptop';

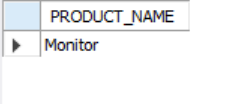


1. Find the Products That Have Never Been Ordered

SELECT P.PRODUCT\_NAME FROM PRODUCT P

LEFT JOIN ORDER\_ITEMS OI ON P.PRODUCT\_ID = OI.PRODUCT\_ID

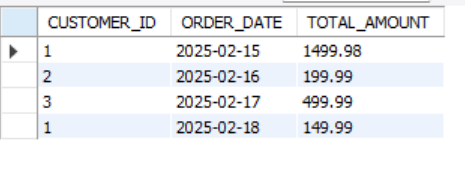
WHERE OI.PRODUCT\_ID IS NULL;



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

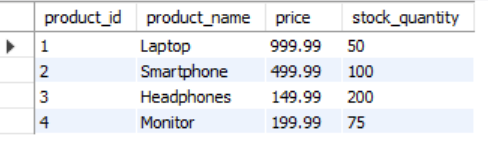
SELECT CUSTOMER\_ID, ORDER\_DATE, TOTAL\_AMOUNT FROM ORDERS

WHERE ORDER\_DATE >= (SELECT MAX(ORDER\_DATE) - 7);



1. Create a view named product\_details that includes all columns from the product table.

CREATE VIEW PRODUCT\_DETAILS AS SELECT \* FROM PRODUCT;



1. 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.

CREATE VIEW ORDER\_SUMMARY AS

SELECT O.ORDER\_ID, O.CUSTOMER\_ID, O.ORDER\_DATE, O.TOTAL\_AMOUNT, S.STATUS\_NAME

FROM ORDERS O

JOIN STATUS S ON O.STATUS\_ID = S.STATUS\_ID;

SELECT \* FROM ORDER\_SUMMARY;

