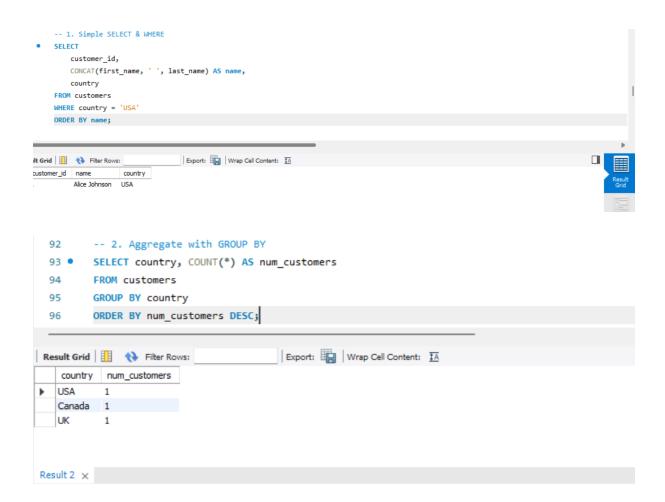
TASK-4 Use SQL queries to extract and analyse data from a database.



```
111
          -- 4. JOIN across three tables
 112 • SELECT
            CONCAT(c.first_name, ' ', c.last_name) AS customer_name,
 113
 114
             p.category,
              SUM(oi.quantity * oi.price) AS total_spent
 115
 116
          FROM order_items oi
          JOIN orders o ON oi.order_id = o.order_id
 117
 118
          JOIN products p ON oi.product_id = p.product_id
          JOIN customers c ON o.customer_id = c.customer_id
 119
           GROUP BY c.customer_id, p.category
 120
 121
           ORDER BY total_spent DESC
           LIMIT 10;
 122
 Export: Wrap Cell Content: IA
     customer_name category total_spent
   Alice Johnson
                   Electronics
                              1799.96
   Bob Smith Books 29.99
        -- 3. JOIN: Find orders with customer info
 100
         o.order id,
         o.order_date,
 101
          CONCAT(c.first_name, ' ', c.last_name) AS customer_name,
          o.status
103
104
      FROM orders o
 105
       INNER JOIN customers c ON o.customer_id = c.customer_id
       WHERE o.order_date > '2023-01-01'
106
107
      ORDER BY o.order_date DESC
108
       LIMIT 10000;
Export: Wrap Cell Content: IA
  order_id order_date customer_name status

        1003
        2025-06-27
        Alice Johnson
        Delivered

        1002
        2025-06-26
        Bob Smith
        Processing

        2025-06-25 Alice Johnson Shipped
 1001
        -- 5. Subquery: Products above average price
 125
 126 • SELECT product_id, product_name, price
 127
      FROM products
 128 ⊝ WHERE price > (
 129
         SELECT AVG(price)
         FROM products
 130
 131
 132
        LIMIT 10000;
 133
| Edit: 🔏 🖶 🖶 | Export/Import: 📳 🗞 | Wrap Cell Content: 🔣
 product_id product_name price
▶ 101
* NULU
         Smartphone 699.99
```

```
-- 6. Subquery in SELECT: Customer total spend
 136 •
          SELECT
 137
            c.customer_id,
 138
             CONCAT(c.first_name, ' ', c.last_name) AS customer_name,
 139
               SELECT SUM(oi.quantity * oi.price)
 141
               FROM orders o
 142
              JOIN order_items oi ON o.order_id = oi.order_id
 143
              WHERE o.customer_id = c.customer_id
          ) AS total_spent
 144
 145
          FROM customers c
 146
          ORDER BY total_spent DESC
 147
         LIMIT 5;
Export: Wrap Cell Content: IA
   customer_id customer_name total_spent
            Alice Johnson 1799.96
Bob Smith 29.99
                                29.99
NULL
   2
 3
                Charlie Brown
SCHEMAS
                               Q Filter objects
e_commerce_db

arrange commerce_db

arrange commerce_db
                                      -- 7. Create VIEW
                              150
                               151 • CREATE VIEW customer_spending AS
                                    SELECT
                               152
                                    c.customer_id,
                              153
                                      CONCAT(c.first_name, ' ', c.last_name) AS customer_name,
                              155
                                       SUM(oi.quantity * oi.price) AS total_spent
                                    FROM customers c
                              156
▼ Views
▼ customer_spending
                              157
                                     JOIN orders o ON c.customer_id = o.customer_id
                               158
                                     JOIN order_items oi ON o.order_id = oi.order_id

    customer_id
    customer_name
    total_spent
                              159
                                     GROUP BY c.customer id;
                              160
    Stored Procedures
                               161
                              Output :
Information ::::::
                             Action Output
           -- 8. Index Optimization
  163 • CREATE INDEX idx_orders_customer ON orders(customer_id);
           -- Run EXPLAIN on a query to check use of index:
  165 • EXPLAIN SELECT * FROM orders WHERE customer_id = 12345;
 Result Grid Filter Rows:
                                          Export: Wrap Cell Content: TA
                                                                                                                                                           id select_type table partitions type possible_keys key key_len ref rows filtered Extra

1 SIMPLE orders GROSS ref idx_orders_customer idx_orders_customer 5 const 1 100.00
 167
          -- 9. Left Join and Right Join
 169
           c.customer id.
          CONCAT(c.first_name, ' ', c.last_name) AS customer_name,
 170
            o.order_id,
         o.order_date
 172
 173
          FROM customers c
          LEFT JOIN orders o ON c.customer_id = o.customer_id
 175
          ORDER BY c.customer id;
 176
```