```sql
/*
    =========================================
    == SUBQUERIES PRACTICE FILE - 30 TASKS ==
    =========================================
*/

USE MyDatabase;


/* =====================================================
   TASK 1
   Find customers whose score is greater than
   the average score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (SELECT AVG(score) FROM Customers);



/* =====================================================
   TASK 2
   Find customers whose score is lower than
   the average score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score < (SELECT AVG(score) FROM Customers);


/* =====================================================
   TASK 3
   Find customer(s) with the highest score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score = (SELECT MAX(score) FROM Customers);


/* =====================================================
   TASK 4
   Find customer(s) with the lowest score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score = (SELECT MIN(score) FROM Customers);


/* =====================================================
   TASK 5
   Show customers whose score is higher than
   the maximum order sales value.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (SELECT MAX(sales) FROM Orders);
```

```sql
/* =====================================================
   TASK 6
   Show customers whose score is lower than
   the minimum order sales value.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score < (SELECT MIN(sales) FROM Orders);



/* =====================================================
   TASK 7
   Find customers who placed orders.
===================================================== */
SELECT first_name
FROM Customers
WHERE id IN (SELECT customer_id FROM Orders);



/* =====================================================
   TASK 8
   Find customers who never placed orders.
===================================================== */
SELECT first_name
FROM Customers
WHERE id NOT IN (SELECT customer_id FROM Orders);



/* =====================================================
   TASK 9
   Find customers who have orders using EXISTS.
===================================================== */
SELECT first_name
FROM Customers c
WHERE EXISTS (
    SELECT 1 FROM Orders o
    WHERE o.customer_id = c.id
);



/* =====================================================
   TASK 10
   Find orders above average sales.
===================================================== */
SELECT order_id, sales
FROM Orders
WHERE sales > (SELECT AVG(sales) FROM Orders);
```

```sql
/* =====================================================
   TASK 11
   Find orders below average sales.
===================================================== */
SELECT order_id, sales
FROM Orders
WHERE sales < (SELECT AVG(sales) FROM Orders);



/* =====================================================
   TASK 12
   Show each customer with the average score.
===================================================== */
SELECT first_name, score,
       (SELECT AVG(score) FROM Customers) AS avg_score
FROM Customers;



/* =====================================================
   TASK 13
   Find customers with score higher than
   the average German customers score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (
    SELECT AVG(score)
    FROM Customers
    WHERE country = 'Germany'
);



/* =====================================================
   TASK 14
   Find customers from USA whose score is
   higher than the global average.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE country = 'USA'
AND score > (SELECT AVG(score) FROM Customers);


/* =====================================================
   TASK 15
   Find customers whose score equals
   the second highest score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score = (
    SELECT MAX(score)
    FROM Customers
    WHERE score < (SELECT MAX(score) FROM Customers)
);
```

```sql
/* =====================================================
   TASK 16
   Find customers whose score equals
   the average score.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score = (SELECT AVG(score) FROM Customers);



/* =====================================================
   TASK 17
   Find customers whose score is above
   the average of customers who placed orders.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (
    SELECT AVG(score)
    FROM Customers
    WHERE id IN (SELECT customer_id FROM Orders)
);



/* =====================================================
   TASK 18
   Find customers who never made sales
   above average sales.
===================================================== */
SELECT first_name
FROM Customers
WHERE id NOT IN (
    SELECT customer_id
    FROM Orders
    WHERE sales > (SELECT AVG(sales) FROM Orders)
);



/* =====================================================
   TASK 19
   Find customers whose score is greater than
   total sales of customer 1.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (
    SELECT SUM(sales)
    FROM Orders
    WHERE customer_id = 1
);
```

```
/* =====================================================
   TASK 20
   Find average total sales per customer.
===================================================== */
SELECT AVG(total_sales)
FROM (
    SELECT SUM(sales) AS total_sales
    FROM Orders
    GROUP BY customer_id
) AS temp;



/* =====================================================
   TASK 21
   Find customers whose score is higher than
   the average total customer sales.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > (
    SELECT AVG(total_sales)
    FROM (
        SELECT SUM(sales) AS total_sales
        FROM Orders
        GROUP BY customer_id
    ) t
);



/* =====================================================
   TASK 22
   Find customers whose score equals
   any order sales value.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score IN (SELECT sales FROM Orders);



/* =====================================================
   TASK 23
   Find customers whose score is not equal
   to any order sales value.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score NOT IN (SELECT sales FROM Orders);
```

```sql
/* =====================================================
   TASK 24
   Increase score by 5 for customers below average.
===================================================== */
UPDATE Customers
SET score = score + 5
WHERE score < (SELECT AVG(score) FROM Customers);



/* =====================================================
   TASK 25
   Decrease score by 5 for customers above average.
===================================================== */
UPDATE Customers
SET score = score - 5
WHERE score > (SELECT AVG(score) FROM Customers);



/* =====================================================
   TASK 26
   Delete customers whose score is below
   minimum order sales.
===================================================== */
DELETE FROM Customers
WHERE score < (SELECT MIN(sales) FROM Orders);



/* =====================================================
   TASK 27
   Find orders placed by customers with high score.
===================================================== */
SELECT order_id
FROM Orders
WHERE customer_id IN (
    SELECT id
    FROM Customers
    WHERE score > (SELECT AVG(score) FROM Customers)
);



/* =====================================================
   TASK 28
   Find customers whose score equals the
   maximum sales made by any order.
===================================================== */
SELECT first_name
FROM Customers
WHERE score = (SELECT MAX(sales) FROM Orders);
```

```sql
/* =====================================================
   TASK 29
   Show customers whose score is greater than
   all order sales.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > ALL (SELECT sales FROM Orders);



/* =====================================================
   TASK 30
   Show customers whose score is greater than
   at least one order sales.
===================================================== */
SELECT first_name, score
FROM Customers
WHERE score > ANY (SELECT sales FROM Orders);
```