a. Using SELECT, WHERE, ORDER BY, GROUP BY

```
SELECT customer_state, COUNT(*) AS total_customers

FROM customers

WHERE customer_state = 'SP'

GROUP BY customer_state

ORDER BY total_customers DESC;
```

b. Using JOINS (INNER, LEFT, RIGHT)

INNER JOIN

```
SELECT c.customer_id, c.customer_city, o.order_id

FROM customers c

INNER JOIN orders o

ON c.customer_id = o.customer_id;
```

RIGHT JOIN

(SQLite does not support RIGHT JOIN directly. Use reverse LEFT JOIN.)

```
SELECT c.customer_id, c.customer_city, o.order_id

FROM orders o

LEFT JOIN customers c

ON c.customer_id = o.customer_id;
```

Subquery

(Get customers who placed at least one order)

```
SELECT customer_id, customer_city

FROM customers

WHERE customer_id IN (

   SELECT customer_id

   FROM orders

);
```

d. Aggregate Functions (SUM, AVG)

(Requires joining with payments or orders)

Total number of customers per state

```
SELECT customer_state, COUNT(customer_id) AS total_customers

FROM customers

GROUP BY customer_state;
```

Average payment by customer state

```
SELECT c.customer_state, AVG(p.payment_value) AS avg_payment

FROM customers c

JOIN orders o ON c.customer_id = o.customer_id

JOIN order_payments p ON o.order_id = p.order_id

GROUP BY c.customer_state;
```

e. Create View for Analysis

```
CREATE VIEW customer_order_summary AS

SELECT c.customer_id,

    c.customer_city,

    c.customer_state,

    COUNT(o.order_id) AS total_orders

FROM customers c

LEFT JOIN orders o

ON c.customer_id = o.customer_id

GROUP BY c.customer_id;
```

To use view:

```
SELECT * FROM customer_order_summary;
```

f. Optimize Queries with Indexes

```
CREATE INDEX idx_customer_id

ON customers(customer_id);
```