Basic Queries

Retrieve all customers who live in a specific state.

```
SELECT first_name, last_name, state
FROM customers
WHERE state = 'California';
```

Find all products with a list price above $500.

```
SELECT product_name, list_price
FROM products
WHERE list_price > 500;
```

List all active staff members along with their store details.

```
SELECT s.first_name, s.last_name, st.store_name
FROM staffs s
JOIN stores st ON s.store_id = st.store_id
WHERE s.active = 1;
```

Intermediate Queries

Find the total quantity of products in stock at each store.

```
SELECT st.store_name, SUM(stk.quantity) AS total_stock
FROM stores st
JOIN stocks stk ON st.store_id = stk.store_id
GROUP BY st.store_name;
```

List the top 5 customers by the number of orders they've placed.

```
SELECT c.first_name, c.last_name, COUNT(o.order_id) AS order_count
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
ORDER BY order_count DESC
LIMIT 5;
```

Find the total revenue generated by each product.

```
SELECT p.product_name, SUM(oi.quantity * oi.list_price * (1 - oi.discount)) AS
total_revenue
FROM products p
JOIN order_items oi ON p.product_id = oi.product_id
GROUP BY p.product_name
ORDER BY total_revenue DESC;
```

Advanced Queries

Identify the store with the highest total sales revenue.

```
SELECT s.store_name, SUM(oi.quantity * oi.list_price * (1 - oi.discount)) AS
total_revenue
FROM stores s
JOIN orders o ON s.store_id = o.store_id
JOIN order_items oi ON o.order_id = oi.order_id
GROUP BY s.store_name
ORDER BY total_revenue DESC
LIMIT 1;
```

List customers who haven't placed any orders.

```
SELECT c.first_name, c.last_name
FROM customers c
LEFT JOIN orders o ON c.customer_id = o.customer_id
WHERE o.order_id IS NULL;
```

Determine the monthly sales trends for the past year.

```
SELECT DATE_FORMAT(o.order_date, '%Y-%m') AS month,
       SUM(oi.quantity * oi.list_price * (1 - oi.discount)) AS monthly_revenue
FROM orders o
JOIN order_items oi ON o.order_id = oi.order_id
WHERE o.order_date >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
GROUP BY DATE_FORMAT(o.order_date, '%Y-%m')
ORDER BY month;
```

Find the most sold product category.

```
SELECT c.category_name, SUM(oi.quantity) AS total_quantity
FROM categories c
JOIN products p ON c.category_id = p.category_id
JOIN order_items oi ON p.product_id = oi.product_id
GROUP BY c.category_name
```

```
ORDER BY total_quantity DESC
LIMIT 1;
```