Task 6: Sales Trend Analysis Using Aggregations (SQLite)

Step 1: Create table

```
CREATE TABLE online_sales (
    order_id INTEGER,
    order_date TEXT,
    product_id INTEGER,
    amount REAL
);

-- Step 2: Insert data
INSERT INTO online_sales VALUES (101, '2024-01-05', 1, 120.0);
INSERT INTO online_sales VALUES (102, '2024-01-10', 2, 80.0);
INSERT INTO online_sales VALUES (103, '2024-02-01', 3, 150.0);
INSERT INTO online_sales VALUES (104, '2024-02-15', 1, 220.0);
INSERT INTO online_sales VALUES (105, '2024-03-03', 2, 90.0);
INSERT INTO online_sales VALUES (106, '2024-03-15', 2, 110.0);
INSERT INTO online_sales VALUES (107, '2024-03-25', 3, 250.0);
INSERT INTO online_sales VALUES (108, '2024-04-04', 1, 300.0);
INSERT INTO online_sales VALUES (109, '2024-04-20', 2, 130.0);
INSERT INTO online_sales VALUES (110, '2024-04-25', 3, 75.0);

-- Step 3: Create view
CREATE VIEW sales_monthly AS
SELECT
    strftime('%Y', order_date) AS year,
    strftime('%m', order_date) AS month,
    SUM(amount) AS monthly_revenue,
    COUNT(DISTINCT order_id) AS order_volume
FROM
    online_sales
GROUP BY
    year, month
ORDER BY
    year, month;

-- Step 4: Query all
SELECT * FROM sales_monthly;
```

```
sqlite> SELECT * FROM sales_monthly;
2024|01|600.0|2
2024|02|1110.0|2
2024|03|1350.0|3
2024|04|1515.0|3
```

```
-- Step 5: Filtered year
SELECT * FROM sales_monthly WHERE year = '2024';
```

```
sqlite> -- Step 5: Filtered year
sqlite> SELECT * FROM sales_monthly WHERE year = '2024';
2024|01|600.0|2
2024|02|1110.0|2
2024|03|1350.0|3
2024|04|1515.0|3
```