

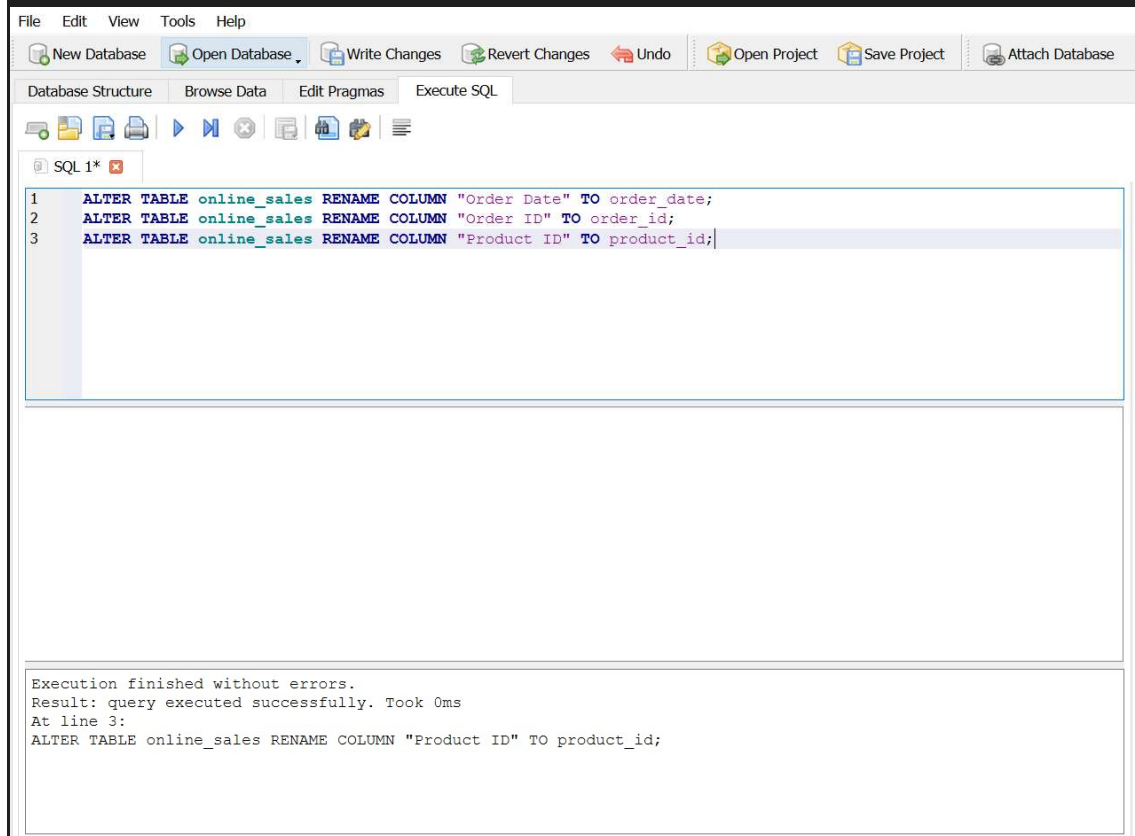
```
-- =====  
-- ONLINE SALES – CLEANING & EDA  
-- =====
```

```
-- -----  
-- STEP 1: Check table structure  
-- -----
```

```
PRAGMA table_info(online_sales);
```

```
-- -----  
-- STEP 2: Fix column names (if needed)  
-- -----
```

```
ALTER TABLE online_sales RENAME COLUMN "Order ID" TO order_id;  
ALTER TABLE online_sales RENAME COLUMN "Order Date" TO order_date;  
ALTER TABLE online_sales RENAME COLUMN "Product ID" TO product_id;
```



```
-- -----  
-- STEP 3: Convert DD-MM-YYYY → YYYY-MM-DD  
-- Example: 11-08-2016 → 2016-08-11  
-- -----
```

```
UPDATE online_sales  
SET order_date =  
    substr(order_date, 7, 4) || '-' ||  
    substr(order_date, 4, 2) || '-' ||
```

```

        substr(order_date, 1, 2)
WHERE order_date LIKE '__-__-____';

-----
-- STEP 4: Convert MM/DD/YYYY → YYYY-MM-DD
-- Example: 06/12/2016 → 2016-06-12
-----

UPDATE online_sales
SET order_date =
    substr(order_date, instr(order_date, '/') + 1, 2) || '-' ||
    substr(order_date, 1, instr(order_date, '/') - 1) || '-' ||
    substr(order_date, length(order_date) - 3, 4)
WHERE order_date LIKE '%/%';

-----
-- STEP 5: Validate clean dates
-----

SELECT
    order_date,
    strftime('%Y', order_date) AS year,
    strftime('%m', order_date) AS month
FROM online_sales
LIMIT 5;

```

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Undo Open Project Save Project Attach Database

Database Structure Browse Data Edit Pragmas Execute SQL

SQL 1*

```

1  SELECT
2      order_date,
3      strftime('%Y', order_date) AS year,
4      strftime('%m', order_date) AS month
5  FROM online_sales
6  LIMIT 5;
7

```

	order_date	year	month
1	2016-08-11	2016	08
2	2016-08-11	2016	08
3	2016-12-06	2016	12
4	2015-11-10	2015	11
5	2015-11-10	2015	11

Execution finished without errors.
Result: 5 rows returned in 12ms
At line 1:
SELECT
 order_date,
 strftime('%Y', order_date) AS year,
 strftime('%m', order_date) AS month
FROM online_sales
LIMIT 5;

```
-- STEP 6: Extract YEAR + MONTH
```

```
SELECT
    strftime('%Y', order_date) AS year,
    strftime('%m', order_date) AS month
FROM online_sales
LIMIT 5;
```

```
-- STEP 7: Monthly Revenue + Order Volume
```

```
SELECT
    strftime('%Y', order_date) AS year,
    strftime('%m', order_date) AS month,
    SUM(amount) AS total_revenue,
    COUNT(DISTINCT order_id) AS total_orders
FROM online_sales
GROUP BY year, month
ORDER BY year, month;
```

The screenshot shows a database IDE interface. The top menu bar includes File, Edit, View, Tools, and Help. Below the menu is a toolbar with icons for New Database, Open Database, Write Changes, Revert Changes, Undo, Open Project, Save Project, and Attach Database. The main window is titled 'SQL 1*' and contains the following SQL query:

```
1 SELECT
2     strftime('%Y', order_date) AS year,
3     strftime('%m', order_date) AS month,
4     SUM(amount) AS total_revenue,
5     COUNT(DISTINCT order_id) AS total_orders
6 FROM online_sales
7 GROUP BY year, month
8 ORDER BY year, month;
```

Below the query editor, the results are displayed in a table with 5 columns: year, month, total_revenue, and total_orders. The table contains 7 rows of data, with the first row showing NULL values for year and month.

	year	month	total_revenue	total_orders
1	NULL	NULL	1409283.5487	2997
2	2014	01	19640.427	36
3	2014	02	11833.618	33
4	2014	03	7159.67	33
5	2014	04	12455.482	29
6	2014	05	15280.411	36
7	2014	06	11927.849	32

Below the table, the execution status is shown: 'Execution finished without errors. Result: 49 rows returned in 63ms. At line 1:'. The SQL query is repeated below this status message.

```
-- STEP 8: Filter by year (example: 2015 only)
```

```
SELECT
    strftime('%Y', order_date) AS year,
    strftime('%m', order_date) AS month,
    SUM(amount) AS total_revenue,
```

```

COUNT(DISTINCT order_id) AS total_orders
FROM online_sales
WHERE strftime('%Y', order_date) = '2015'
GROUP BY year, month
ORDER BY month;

```

The screenshot shows a SQL IDE interface with a menu bar (File, Edit, View, Tools, Help) and a toolbar. The main window displays a SQL query in a text editor, which is identical to the one in the first block. Below the editor, a table of results is shown, containing 7 rows of data for the year 2015. The columns are year, month, total_revenue, and total_orders. Below the table, a status bar indicates that the execution finished without errors and that 12 rows were returned in 24ms.

	year	month	total_revenue	total_orders
1	2015	01	17701.6864	32
2	2015	02	13018.315	33
3	2015	03	12207.4066	34
4	2015	04	11963.696	28
5	2015	05	10483.482	46
6	2015	06	13112.6482	35
7	2015	07	10706.72	34

Execution finished without errors.
Result: 12 rows returned in 24ms
At line 1:
SELECT
 strftime('%Y', order_date) AS year,
 strftime('%m', order_date) AS month,
 SUM(amount) AS total_revenue,
 COUNT(DISTINCT order_id) AS total_orders

```
-- STEP 9: Top 5 Highest Revenue Months
```

```

SELECT
    strftime('%Y-%m', order_date) AS month,
    SUM(amount) AS revenue
FROM online_sales
GROUP BY month
ORDER BY revenue DESC
LIMIT 5;

```

FileEditViewToolsHelp

New DatabaseOpen DatabaseWrite ChangesRevert ChangesUndoOpen ProjectSave ProjectAttach Database

Database StructureBrowse DataEdit PragmasExecute SQL

SQL 1*

1SELECT

2 strftime('%Y-%m', order_date) AS month,

3 SUM(amount) AS revenue

4FROM online_sales

5GROUP BY month

6ORDER BY revenue DESC

7LIMIT 5;

8

	month	revenue
1	NULL	1409283.5487
2	2016-02	43298.735
3	2017-02	36988.456
4	2017-01	31839.912
5	2017-03	26899.473

Execution finished without errors.
Result: 5 rows returned in 27ms
At line 1:
SELECT
 strftime('%Y-%m', order_date) AS month,
 SUM(amount) AS revenue
FROM online_sales
GROUP BY month

-- =====
-- END OF SCRIPT
-- =====