

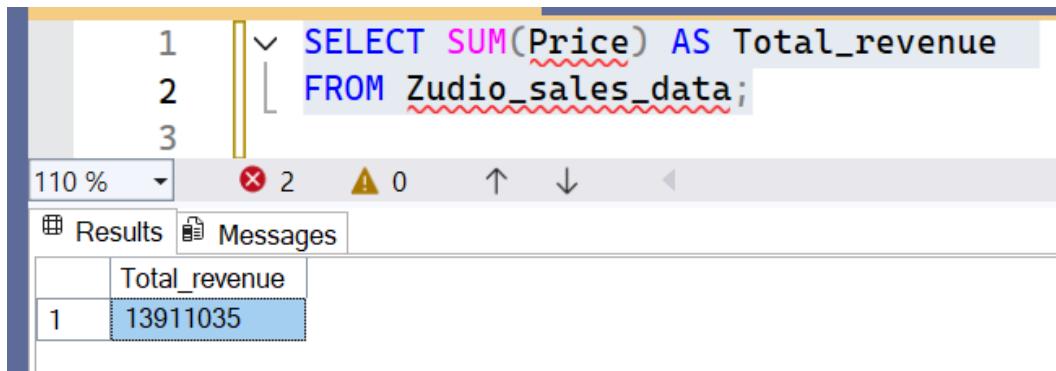
ZUDIO SALES SQL QUERIES

A. KPI's

We need to analyze key indicators for our Zudio sales data to gain insights into our business performance. Specifically, we want to calculate the following metrics:

1. Total Sales Revenue :

```
SELECT SUM(Price) AS Total_revenue  
FROM Zudio_sales_data;
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

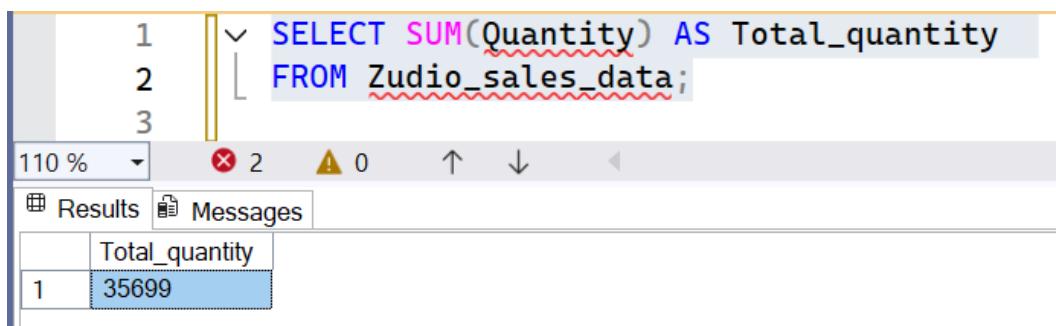
```
1  SELECT SUM(Price) AS Total_revenue  
2  FROM Zudio_sales_data;  
3
```

The results pane shows a single row of data:

Total_revenue
13911035

2. Total Quantity Sold :

```
SELECT SUM(Quantity) AS Total_quantity  
FROM Zudio_sales_data;
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

```
1  SELECT SUM(Quantity) AS Total_quantity  
2  FROM Zudio_sales_data;  
3
```

The results pane shows a single row of data:

Total_quantity
35699

3. Total Number of Orders :

```
SELECT  
    COUNT(*) AS Total_Orders  
FROM Zudio_sales_data;
```

The screenshot shows a SQL query results window. On the left, there is a vertical list of line numbers from 95 to 98. The main area contains the SQL code for calculating the total number of orders. The result set is displayed in a table with one row, showing a value of 7899 for the column 'Total_Orders'. The status bar at the bottom indicates 'No issues found'.

	Total_Orders
1	7899

4. Average Order Value (AOV) :

```
SELECT  
    SUM(Price) * 1.0 / COUNT(*) AS Avg_Order_Value  
FROM Zudio_sales_data;
```

The screenshot shows a SQL query results window. On the left, there is a vertical list of line numbers from 101 to 104. The main area contains the SQL code for calculating the average order value. The result set is displayed in a table with one row, showing a value of 1761.11343208001 for the column 'Avg_Order_Value'. The status bar at the bottom indicates 'No issues found'.

	Avg_Order_Value
1	1761.11343208001

5. Average Quantity per Order :

```
SELECT
    SUM(Quantity) * 1.0 / COUNT(*) AS Avg_Quantity_per_Order
FROM Zudio_sales_data;
```

The screenshot shows a code editor with a numbered left margin (107 to 110) and a right margin (133 %). The main area contains a SELECT statement. Below the code is a status bar with a green checkmark and the text "No issues found". At the bottom, there are tabs for "Results" and "Messages", and a results table with one row:

	Avg_Quantity_per_Order
1	4.519432839599

B. PIVOT TABLES AND PIVOT CHARTS QUERIES

1. Total Sales Trend (Monthly)

```
SELECT FORMAT(Order_Date, 'yyyy-MM') AS Month,
        SUM(Price) AS Total_Sales
    FROM Zudio_sales_data
GROUP BY FORMAT(Order_Date, 'yyyy-MM')
ORDER BY Month;
```

The screenshot shows a code editor with a numbered left margin (4 to 10) and a right margin (110 %). The main area contains a SELECT statement with various clauses. Below the code is a status bar with a green checkmark and the text "No issues found". At the bottom, there are tabs for "Results" and "Messages", and a results table with 12 rows:

	Month	Total_Sales
1	2024-01	1256901
2	2024-02	1219689
3	2024-03	1257225
4	2024-04	1228644
5	2024-05	1262029
6	2024-06	1203358
7	2024-07	1217543
8	2024-08	1293240
9	2024-09	1166772
10	2024-10	1343307
11	2024-11	1313258
12	2024-12	149069

2. Sales by Product Category

```
SELECT
    Category,
    SUM(Quantity) AS Total_Qty_Sold,
    SUM(Price) AS Total_Sales
FROM Zudio_sales_data
GROUP BY Category
ORDER BY Total_Sales DESC;
```

The screenshot shows a SQL editor interface with the following details:

- Code Area:** Displays the SQL query with line numbers 13 to 20 on the left. The code uses color-coded syntax highlighting.
- Status Bar:** Shows "110 %" and "No issues found".
- Results Tab:** Active tab, showing a table with three rows of data.
- Messages Tab:** Inactive tab.
- Table Data:**

	Category	Total_Qty_Sold	Total_Sales
1	Kids	12013	4707479
2	Men	11727	4661546
3	Women	11959	4542010

3. Top 5 Best-Selling Products

```
SELECT TOP 5
    Clothing_Type,
    SUM(Quantity) AS Total_Quantity
FROM Zudio_sales_data
GROUP BY Clothing_Type
ORDER BY Total_Quantity DESC;
```

The screenshot shows a SQL query being run in a database environment. The query is:

```
22
23     SELECT TOP 5
24         Clothing_Type,
25             SUM(Quantity) AS Total_Quantity
26     FROM Zudio_sales_data
27     GROUP BY Clothing_Type
28     ORDER BY Total_Quantity DESC;
29
```

The results are displayed in a table:

	Clothing_Type	Total_Quantity
1	Dresses	3375
2	Skirts	3305
3	Hoodies	3294
4	T-shirts	3288
5	Jackets	3275

4. Sales by State

```
SELECT
    State,
    SUM(Price) AS Total_Sales
FROM Zudio_sales_data
GROUP BY State
ORDER BY Total_Sales DESC;
```

The screenshot shows a SQL query being run in a database environment. The code is as follows:

```
31
32     SELECT
33         State,
34         SUM(Price) AS Total_Sales
35     FROM Zudio_sales_data
36     GROUP BY State
37     ORDER BY Total_Sales DESC;
```

The results tab displays the following data:

	State	Total_Sales
1	Rajasthan	1803356
2	Uttar Pradesh	1785224
3	West Bengal	1762483
4	Delhi	1758945
5	Gujarat	1716392
6	Tamil Nadu	1705167
7	Maharashtra	1689805
8	Karnataka	1689663

The messages tab indicates "No issues found".

5. Contribution by Product Type (Tshirt, Kurti, Jeans, etc.)

```
SELECT
    Clothing_Type,
    SUM(Price) AS Total_Sales
FROM Zudio_sales_data
GROUP BY Clothing_Type
ORDER BY Total_Sales DESC;
```

The screenshot shows a SQL query being run in a database environment. The query is:

```
41  SELECT
42      Clothing_Type,
43          SUM(Price) AS Total_Sales
44  FROM Zudio_sales_data
45  GROUP BY Clothing_Type
46  ORDER BY Total_Sales DESC;
47
```

The results tab displays a table with the following data:

	Clothing_Type	Total_Sales
1	Dresses	1320479
2	T-shirts	1301328
3	Skirts	1293344
4	Jackets	1288330
5	Pants	1287756
6	Shirts	1274970
7	Sweaters	1258260
8	Tops	1238021
9	Jeans	1218038
10	Shoes	1215784
11	Hoodies	1214725

6. Average Billing Amount Per Customer

```
SELECT
    AVG(Price) AS Avg_Billing_Amount
FROM Zudio_sales_data;
```

A screenshot of a SQL query results window. The query is:

```
49
50
51
52
      SELECT
        AVG(Price) AS Avg_Billing_Amount
      FROM Zudio_sales_data;
```

The status bar at the bottom says "No issues found". The results table has one row:

	Avg_Billing_Amount
1	1761.11343208001

7. Profit Analysis

```
SELECT
    Clothing_Type,
    SUM(Sales_Profit) AS Profit
FROM Zudio_sales_data
GROUP BY Clothing_Type
ORDER BY Profit DESC;
```

A screenshot of a SQL query results window. The query is:

```
74
75
76
77
78
79
80
      SELECT
        Clothing_Type,
        SUM(Sales_Profit) AS Profit
      FROM Zudio_sales_data
      GROUP BY Clothing_Type
      ORDER BY Profit DESC;
```

The status bar at the bottom says "No issues found". The results table shows the top 11 clothing types and their profits:

	Clothing_Type	Profit
1	Dresses	2373512.40209961
2	Skirts	2320703.19700623
3	Pants	2316501.20129395
4	Sweaters	2315042.80104065
5	Jackets	2304795.59646606
6	Shirts	2294617.59840393
7	Hoodies	2289815.20053101
8	T-shirts	2278154.39254761
9	Tops	2209604.79919434
10	Jeans	2200245.59925842
11	Shoes	2142140.39752197

8. Stored Type Contribution

```
SELECT
    Store_Type,
    SUM(Price) AS Total_Sales
FROM Zudio_sales_data
GROUP BY Store_Type;
```

The screenshot shows a SQL editor interface with the following details:

- Code Area:** Displays the SQL query with line numbers 83 to 89 on the left. The query selects total sales by store type from the 'Zudio_sales_data' table.
- Status Bar:** Shows "83 %", a green checkmark icon, and the text "No issues found".
- Results Tab:** Active tab, showing a table with two rows of data.
- Table Data:**

	Store_Type	Total_Sales
1	Rented	6987234
2	Owned	6923801