**SQL SALES PROJECT REPORT:**

**TOOLS**:Sql online server,data collection from kuggle

**1. Table Creation**

CREATE TABLE retail\_sales (

transactions\_id INT PRIMARY KEY,

sale\_date DATETIME,

sale\_time VARCHAR(8),

customer\_id INT,

gender VARCHAR(6),

age INT,

category VARCHAR(11),

quantity INT,

price\_per\_unit INT,

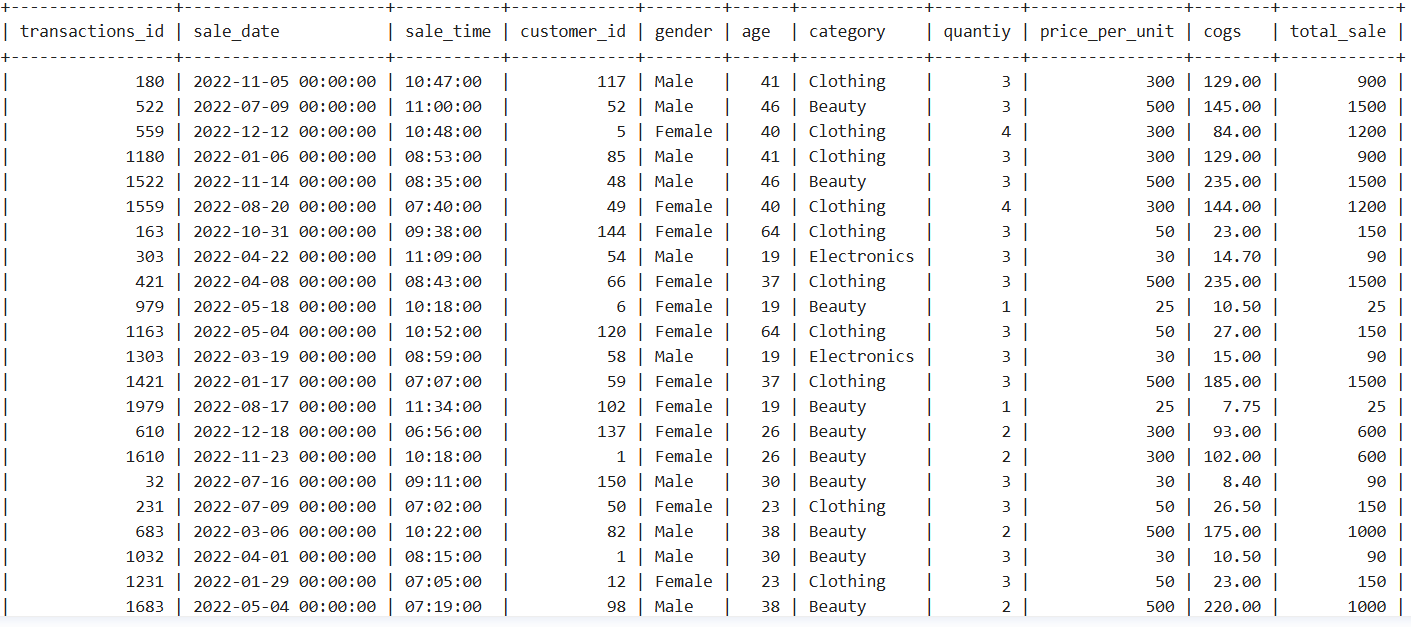
cogs NUMERIC(5, 2),

total\_sale INT

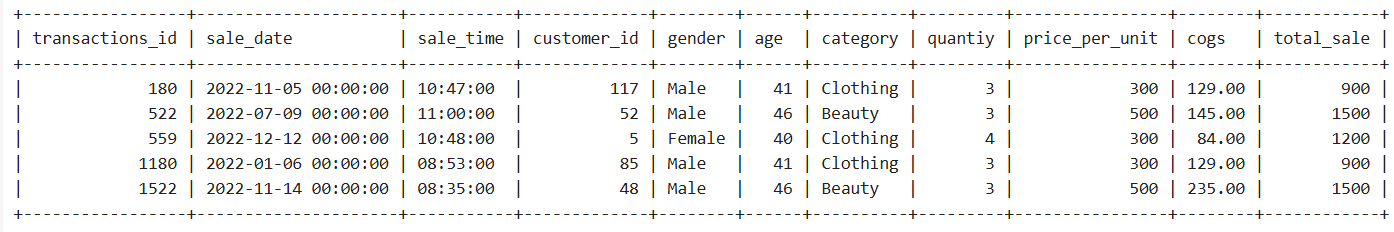
);

**2. Select All Data**

**Select \* from retail sales;**



Select \* from limit 5;

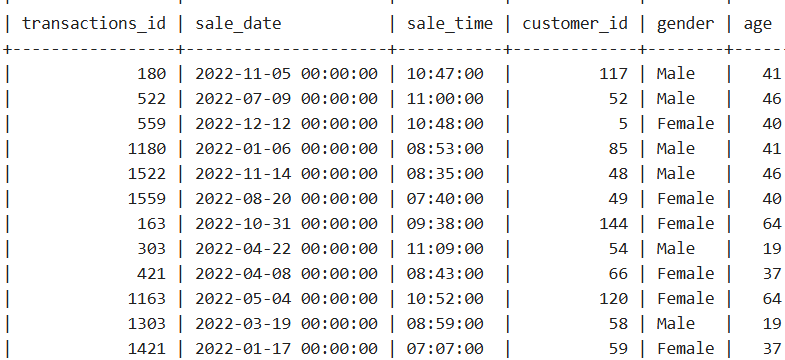


**3. SELECT with WHERE, ORDER BY, GROUP BY**

* Find all sales where quantity > 2, ordered by total\_sale descending:

SELECT \* FROM retail\_sales

WHERE quantity > 2;

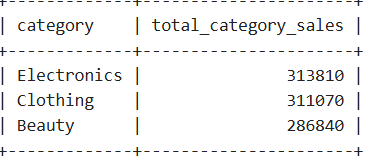


SELECT category, SUM(total\_sale) AS total\_category\_sales

FROM retail\_sales

GROUP BY category

ORDER BY total\_category\_sales DESC;



**4. JOINS**

CREATE TABLE customers (

customer\_id INT PRIMARY KEY,

customer\_name VARCHAR(50),

city VARCHAR(50)

);

* **INNER JOIN** retail\_sales and customers on customer\_id:

SELECT r.transactions\_id, c.customer\_name, r.total\_sale

FROM retail\_sales r

INNER JOIN customers c ON r.customer\_id = c.customer\_id;

* **LEFT JOIN** to get all sales even if customer info is missing:

SELECT r.transactions\_id, c.customer\_name, r.total\_sale

FROM retail\_sales r

LEFT JOIN customers c ON r.customer\_id = c.customer\_id;

* **RIGHT JOIN** to get all customers, even if they have no sales:

SELECT r.transactions\_id, c.customer\_name, r.total\_sale

FROM retail\_sales r

RIGHT JOIN customers c ON r.customer\_id = c.customer\_id;

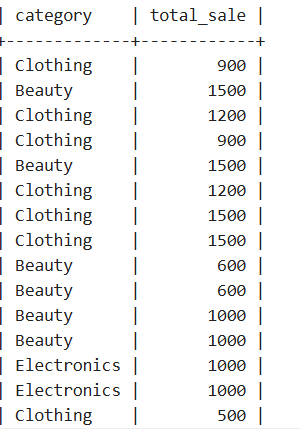
**5. Subqueries**

* Find transactions where total\_sale is greater than average sale:

SELECT sales,category

FROM retail\_sales

WHERE total\_sale > (SELECT AVG(total\_sale) FROM retail\_sales);



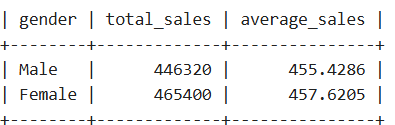
**6. Aggregate Functions**

* Sum and Average total\_sale by gender:

SELECT gender, SUM(total\_sale) AS total\_sales, AVG(total\_sale) AS average\_sales

FROM retail\_sales

GROUP BY gender;



**7. Views**

Create a view for total sales per category:

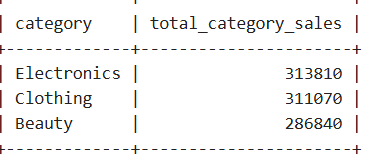
CREATE VIEW category\_sales AS

SELECT category, SUM(total\_sale) AS total\_category\_sales

FROM retail\_sales

GROUP BY category;

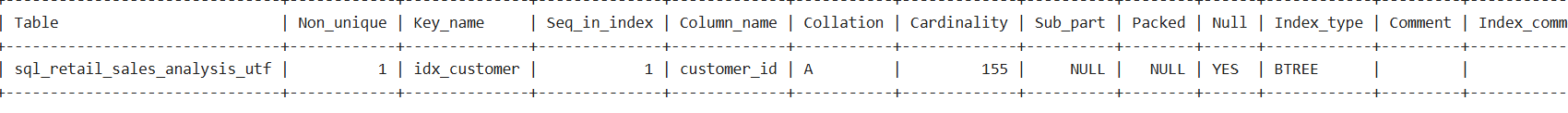
SELECT \* FROM category\_sales ORDER BY total\_category\_sales DESC;



**8. Indexes for Optimization**

Create an index on customer\_id and category to speed up queries:

CREATE INDEX idx\_customer ON retail\_sales(customer\_id);



**Conclusion**:

This report demonstrates comprehensive data analysis using SQL on a retail sales dataset. The analysis included foundational SQL operations such as SELECT, WHERE, ORDER BY, and GROUP BY, as well as more advanced techniques including JOINs, subqueries, aggregate functions (SUM, AVG), views, and indexing.

From the queries, we derived valuable insights such as:

* Gender-based sales performance, showing whether male or female customers contributed more to total sales.
* Category-wise sales analysis, identifying the most popular and profitable product categories.
* Use of indexing to optimize query performance for faster retrieval of key business metrics.

NOTE:

I sdon’t have SQL installed on my system, so I used an online SQL server instead.

