Rohit Ghodke

SQL Portfolio Project

Maven Analytics Mexico Toy Sales

Introduction:

This portfolio project aims to demonstrate my expertise in SQL BY analyzing data using the **Maven Analytics Mexico Toy Sales** dataset. The dataset provides valuable insights into toy sales in Mexico, allowing for a comprehensive examination of sales trends, customer behaviour, and product performance. BY leveraging SQL, I will EXTRACT, transform, and analyze the dataset to uncover key findings and trends that can inform business decisions.

Throughout this project, I will showcase my ability to write SQL queries effectively, manipulate data to EXTRACT meaningful insights, and present findings in a clear and concise manner. BY analyzing the Mexico Toy Sales dataset, I aim to gain insights into factors influencing toy sales, identify top-selling products, understand customer preferences, and explore opportunities for improving sales performance.

For this project, I utilized **Oracle SQL Developer**, a powerful tool designed for database development and management using Oracle Database. With its user-friendly interface and robust features, Oracle SQL Developer enabled me to write, execute, and optimize SQL queries efficiently.

The dataset consists of several tables containing information about **Products, Inventory, Stores, ales, and Calendar dates.**

SUMMARY

Sales Data Observations:

- Year 2023 had missing data for October to December.
- Excluding October to December 2023 has generated more sales than 2022 for the months January to September.

2022 (Jan-Sep)	297,055 units
2023 (Jan-Sep):	408,417 units

Percentage Increase in sales from 2022 to 2023: ((408,417 - 297,055) / 297,055) * 100 ≈ 37.47%

- Forecasted Sales for Oct-Dec 2023:
  After forecasting sales for October to December 2023, the data is as follows:

2022 (Jan-Dec):	420,845 units
2023 (Jan-Dec):	546,457 units

Percentage growth in sales from2022 to 2023: ((546,457 - 420,845) / 420,845) * 100 ≈ 29.80%

- Monthly Sales Growth in 2023:
  Using the predicted data, 2023 monthly sales surpassed 2022's, with growth ranging between 27% and 50%.

Product Categories :

Based on the project data, the product categories include toys, arts and crafts, electronics, games, and sports and outdoors.

- In 2022, the top-selling product category generating the most profit was toys, with 141,345 units sold.
- In 2023, the top-selling product category generating the most profit was arts and crafts, with 203,062 units sold.

Sales by store location:

- In 2022, the Downtown location recorded the highest sales, with toys being their top-selling product category.
- In 2023, the Downtown location again led in sales, with arts and crafts emerging as their best-selling product category.

Store Location Analysis:

Downtown:

- It has the highest number of stores, sales, and profit among all locations.
- However, the profit percentage is comparatively lower in Downtown.

Airport:

- Despite having the fewest number of stores, sales, and profit are higher.
- The profit percentage is the highest at the Airport location.

CITY	STORES
Ciudad de Mexico	FOUR
Hermosillo	THREE
Toluca	TWO
Villahermosa	ONE

Forecast Value

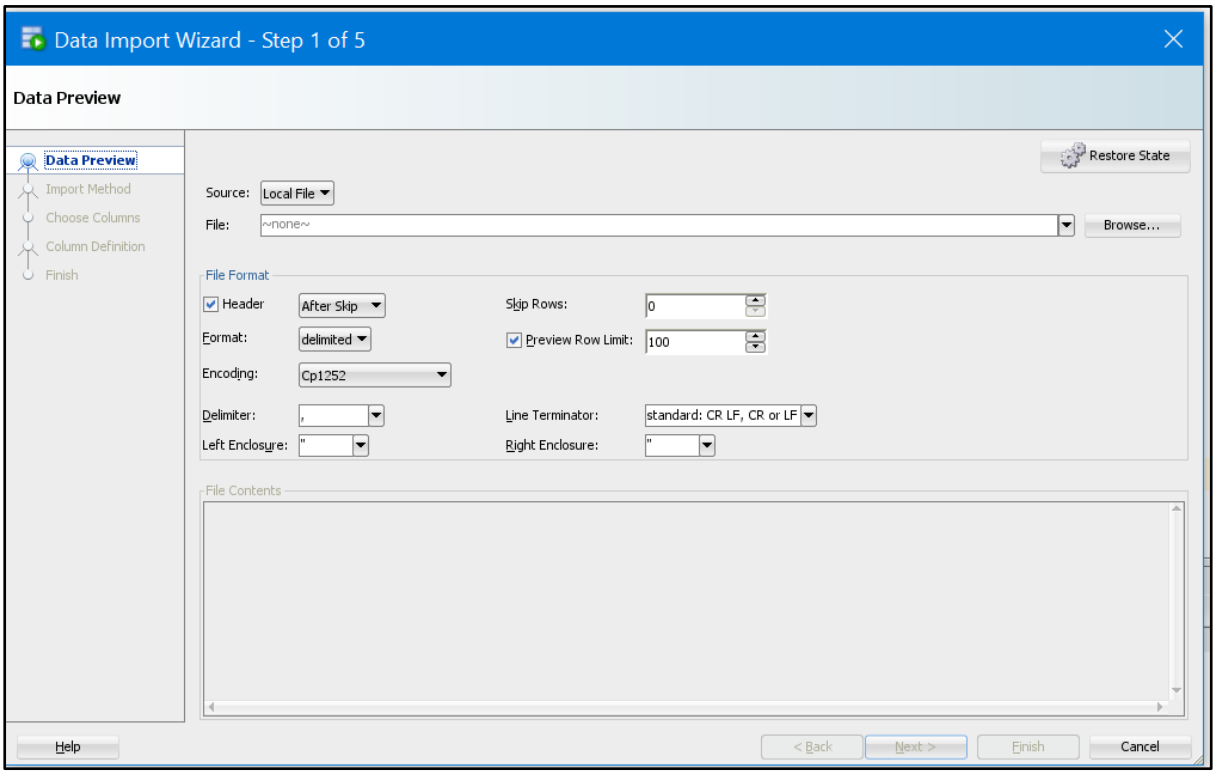
After forecasting sales for October to December 2023, the projected total sales for the entire year are as follows:

Total sales in 2022: 420,845 units

Total sales in 2023: 546,457 units

SQL CODE

First, all the dataset files were imported into Oracle SQL Developer for analysis.



Data Cleaning:

I verified the presence of null values in all tables, including the Sales table. No null values were found in any of the tables.

```
SELECT  * FROM A_CALENDAR
WHERE DATE_ IS NULL;
```

```
SELECT  * FROM A_INVENTORY
WHERE STORE_ID IS NULL OR PRODUCT_ID IS NULL OR STOCK_ON_HAND IS NULL;
```

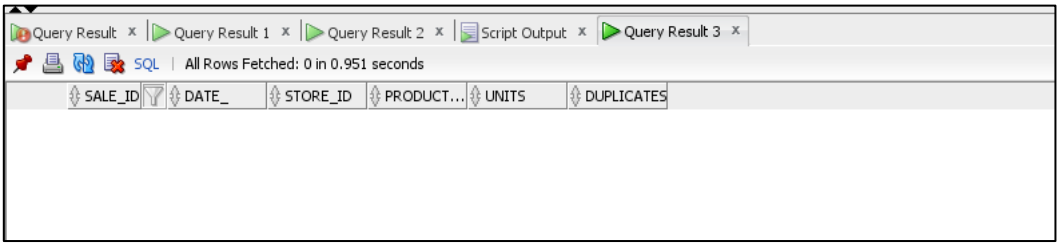
```
SELECT  * FROM A_PRODUCTS
WHERE PRODUCT_ID IS NULL OR product_name IS NULL OR product_cost IS NULL OR
product_category IS NULL OR product_price IS NULL;
```

```
SELECT  * FROM A_SALES
WHERE sale_id IS NULL OR date_ IS NULL OR product_id IS NULL OR store_id IS NULL OR
units IS NULL;
```

- I conducted a thorough check for duplicate values in all the tables. This ensures the integrity of the data and confirms that each transaction is uniquely recorded without any duplications.

Example:

```
SELECT  sale_id, date_, store_id, product_id, units, COUNT(*) AS DUPLICATES
FROM a_sales
GROUP BY sale_id, date_, store_id, product_id, units
HAVING COUNT(*)>1;
```



- To ensure referential integrity between the Sales and Stores tables, I executed a query to identify any records in the Sales table where the Store_ID does not exist in the Stores table. This helps in identifying any discrepancies or inconsistencies in the data that might violate the defined relationships between these tables.

```
--Ensuring Referential Integrity

SELECT  * FROM A_SALES
WHERE STORE_ID NOT IN (SELECT  STORE_ID FROM A_STORE);
```

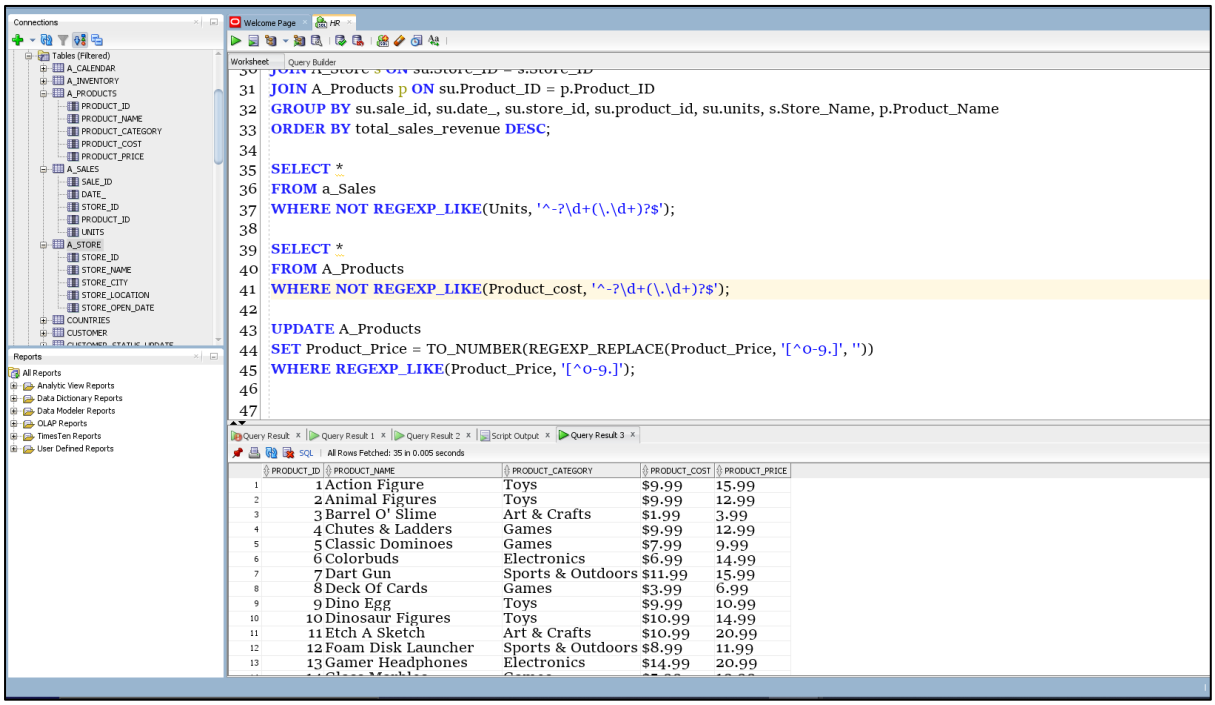
- Then Check whether your data is correctly imported or not.
  After importing the data, I encountered an error related to the presence of dollar signs ('$'). This suggests that there might be issues with the data import process or with the format of the data itself. I will need to check the data to ensure it is correctly imported and address any issues with the '$' signs.

Check using :

```
SELECT  *
FROM a_Sales
WHERE NOT REGEXP_LIKE(Units, '^-?\d+(\.\d+)?$');
```

Update the value :

```
UPDATE A_Products
SET Product_cost = TO_NUMBER (REGEXP_REPLACE (Product_cost, '[^0-9.]', ''))
WHERE REGEXP_LIKE(Product_cost, '[^0-9.]');
```



Joining Tables:

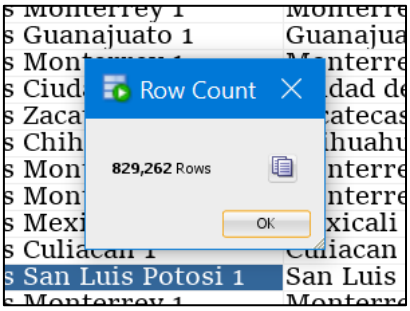
- I combined three tables - Products, Sales, and Stores - by joining them based on a common column present in each table and store in into one table, Joining tables allows for the creation of relationships between related data, ensuring data integrity and consistency. simplified queries, enhanced efficiency, comprehensive analysis, and better data organization and store in into one table.

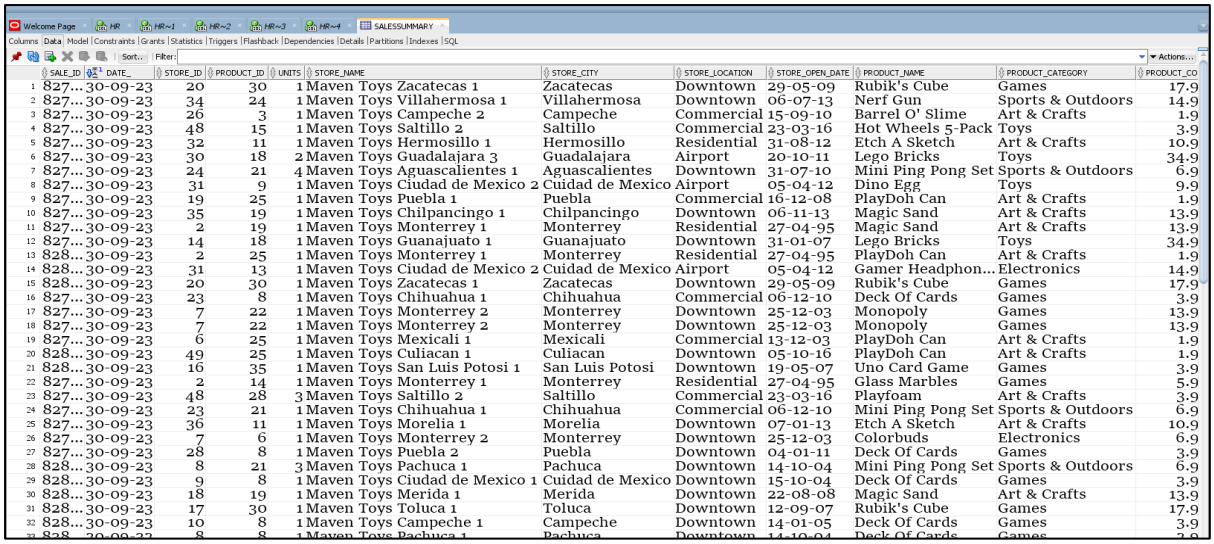
```
CREATE TABLE SalesSummary (
  sale_id INT,
  date_ DATE,
  Store_ID INT,
  Product_ID INT,
  Units INT,
  Store_Name VARCHAR (255),
  store_city VARCHAR (255),
  store_location VARCHAR (255),
```

```
    store_open_date DATE,
    Product_Name VARCHAR (255),
    PRODUCT_CATEGORY VARCHAR (255),
    product_cost DECIMAL (10, 2),
    product_price DECIMAL (10, 2),
    total_sales_revenue DECIMAL (10, 2));


INSERT INTO SalesSummary (sale_id, date_, Store_ID, Product_ID, Units,
                Store_Name, store_city, store_location, store_open_date,
                Product_Name, PRODUCT_CATEGORY, product_cost, product_price,
total_sales_revenue)
SELECT  su.sale_id, su.date_, su.Store_ID, su.Product_ID, su.Units,
    s.Store_Name, s.store_city, s.store_location, s.store_open_date,
    p.Product_Name, p.PRODUCT_CATEGORY, p.product_cost, p.product_price,
    SUM(su.Units * p.Product_Price) AS total_sales_revenue
FROM a_Sales su
JOIN A_Store s ON su.Store_ID = s.Store_ID
JOIN A_Products p ON su.Product_ID = p.Product_ID
GROUP BY su.sale_id, su.date_, su.Store_ID, su.Product_ID, su.Units,
      s.Store_Name, s.store_city, s.store_location, s.store_open_date,
      p.Product_Name, p.PRODUCT_CATEGORY, p.product_cost, p.product_price;
```

- After joining the tables, verify the rows of the resulting table to ensure they match the original data

  In Oracle SQL Developer, you can easily count rows by right-clicking on any value and selecting the "Count Rows" option. This provides a quick way to obtain the total number of rows in a table or the number of rows that meet specific criteria.

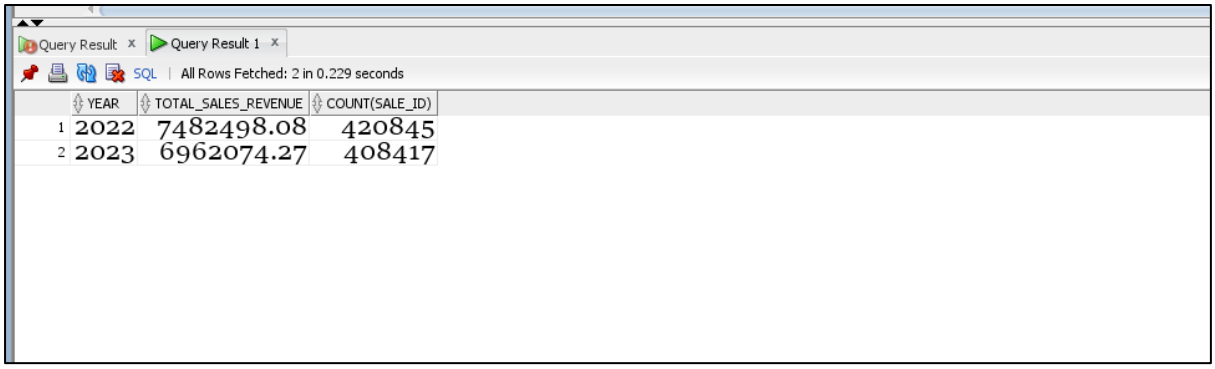




SALE_ID	DATE_	STORE_ID	PRODUCT_ID	UNITS	STORE_NAME	STORE_CITY	STORE_LOCATION	STORE_OPEN_DATE	PRODUCT_NAME	PRODUCT_CATEGORY	PRODUCT_CO
1 827...	30-09-23	20	30	1	Maven Toys Zacatecas 1	Zacatecas	Downtown	29-05-09	Rubik's Cube	Games	17.9
2 827...	30-09-23	34	24	1	Maven Toys Villahermosa 1	Villahermosa	Downtown	06-07-13	Nerf Gun	Sports & Outdoors	14.9
3 827...	30-09-23	26	3	1	Maven Toys Campeche 2	Campeche	Commercial	15-09-10	Barrel O' Slime	Art & Crafts	1.9
4 827...	30-09-23	48	15	1	Maven Toys Saltillo 2	Saltillo	Commercial	23-03-16	Hot Wheels 5-Pack	Toys	3.9
5 827...	30-09-23	32	11	1	Maven Toys Hermosillo 1	Hermosillo	Residential	31-08-12	Etch A Sketch	Art & Crafts	10.9
6 827...	30-09-23	30	18	2	Maven Toys Guadalajara 3	Guadalajara	Airport	20-10-11	Lego Bricks	Toys	34.9
7 827...	30-09-23	24	21	4	Maven Toys Aguascalientes 1	Aguascalientes	Downtown	31-07-10	Mini Ping Pong Set	Sports & Outdoors	6.9
8 827...	30-09-23	31	9	1	Maven Toys Ciudad de Mexico 2	Cuidad de Mexico	Airport	05-04-12	Dino Egg	Toys	9.9
9 827...	30-09-23	19	25	1	Maven Toys Puebla 1	Puebla	Commercial	16-12-08	PlayDoh Can	Art & Crafts	1.9
10 827...	30-09-23	35	19	1	Maven Toys Chilpancingo 1	Chilpancingo	Downtown	06-11-13	Magic Sand	Art & Crafts	13.9
11 827...	30-09-23	2	19	1	Maven Toys Monterrey 1	Monterrey	Residential	27-04-95	Magic Sand	Art & Crafts	13.9
12 827...	30-09-23	14	18	1	Maven Toys Guanajuato 1	Guanajuato	Downtown	31-01-07	Lego Bricks	Toys	34.9
13 828...	30-09-23	2	25	1	Maven Toys Monterrey 1	Monterrey	Residential	27-04-95	PlayDoh Can	Art & Crafts	1.9
14 828...	30-09-23	31	13	1	Maven Toys Ciudad de Mexico 2	Cuidad de Mexico	Airport	05-04-12	Gamer Headphon...	Electronics	14.9
15 828...	30-09-23	20	30	1	Maven Toys Zacatecas 1	Zacatecas	Downtown	29-05-09	Rubik's Cube	Games	17.9
16 827...	30-09-23	23	8	1	Maven Toys Chihuahua 1	Chihuahua	Commercial	06-12-10	Deck Of Cards	Games	3.9
17 827...	30-09-23	7	22	1	Maven Toys Monterrey 2	Monterrey	Downtown	25-12-03	Monopoly	Games	13.9
18 827...	30-09-23	7	22	1	Maven Toys Monterrey 2	Monterrey	Downtown	25-12-03	Monopoly	Games	13.9
19 827...	30-09-23	6	25	1	Maven Toys Mexicali 1	Mexicali	Commercial	13-12-03	PlayDoh Can	Art & Crafts	1.9
20 828...	30-09-23	49	25	1	Maven Toys Culiacan 1	Culiacan	Downtown	05-10-16	PlayDoh Can	Art & Crafts	1.9
21 828...	30-09-23	16	35	1	Maven Toys San Luis Potosi 1	San Luis Potosi	Downtown	19-05-07	Uno Card Game	Games	3.9
22 827...	30-09-23	2	14	1	Maven Toys Monterrey 1	Monterrey	Residential	27-04-95	Glass Marbles	Games	5.9
23 827...	30-09-23	48	28	3	Maven Toys Saltillo 2	Saltillo	Commercial	23-03-16	Playfoam	Art & Crafts	3.9
24 827...	30-09-23	23	21	1	Maven Toys Chihuahua 1	Chihuahua	Commercial	06-12-10	Mini Ping Pong Set	Sports & Outdoors	6.9
25 827...	30-09-23	36	11	1	Maven Toys Morelia 1	Morelia	Downtown	07-01-13	Etch A Sketch	Art & Crafts	10.9
26 827...	30-09-23	7	6	1	Maven Toys Monterrey 2	Monterrey	Downtown	25-12-03	Colorbuds	Electronics	6.9
27 827...	30-09-23	28	8	1	Maven Toys Puebla 2	Puebla	Downtown	04-01-11	Deck Of Cards	Games	3.9
28 828...	30-09-23	8	21	3	Maven Toys Pachuca 1	Pachuca	Downtown	14-10-04	Mini Ping Pong Set	Sports & Outdoors	6.9
29 828...	30-09-23	9	8	1	Maven Toys Ciudad de Mexico 1	Cuidad de Mexico	Downtown	15-10-04	Deck Of Cards	Games	3.9
30 828...	30-09-23	18	19	1	Maven Toys Merida 1	Merida	Downtown	22-08-08	Magic Sand	Art & Crafts	13.9
31 828...	30-09-23	17	30	1	Maven Toys Toluca 1	Toluca	Downtown	12-09-07	Rubik's Cube	Games	17.9
32 828...	30-09-23	10	8	1	Maven Toys Campeche 1	Campeche	Downtown	14-01-05	Deck Of Cards	Games	3.9

In most cases, the first step is to calculate the total sales for the specified time periods or categories.

```
SELECT  EXTRACT(YEAR FROM date_) AS YEAR,
    SUM(TOTAL_SALES_REVENUE) AS TOTAL_SALES_REVENUE,
    COUNT(SALE_ID)
FROM SalesSummary
GROUP BY EXTRACT(YEAR FROM date_)
ORDER BY EXTRACT(YEAR FROM date_);
```



Query Result | Query Result 1

SQL | All Rows Fetched: 2 in 0.229 seconds

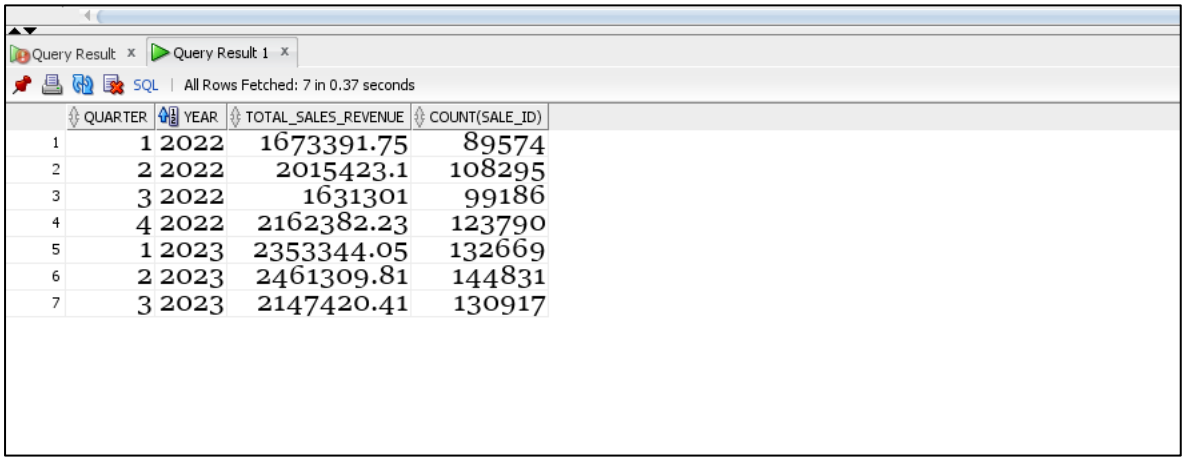
YEAR	TOTAL_SALES_REVENUE	COUNT(SALE_ID)
1 2022	7482498.08	420845
2 2023	6962074.27	408417

- Based on the total sales generated per year, I wanted to explore why 2022 did better than 2023

Then, I checked the sales data quarterly.

```
SELECT  CEIL(EXTRACT(MONTH FROM date_) / 3) AS QUARTER,
    EXTRACT(YEAR FROM date_) AS YEAR,SUM(TOTAL_SALES_REVENUE) AS
TOTAL_SALES_REVENUE,
    COUNT(SALE_ID)
FROM SalesSummary
GROUP BY EXTRACT(YEAR FROM date_), CEIL(EXTRACT(MONTH FROM date_) / 3)
```

```
ORDER BY YEAR;
```



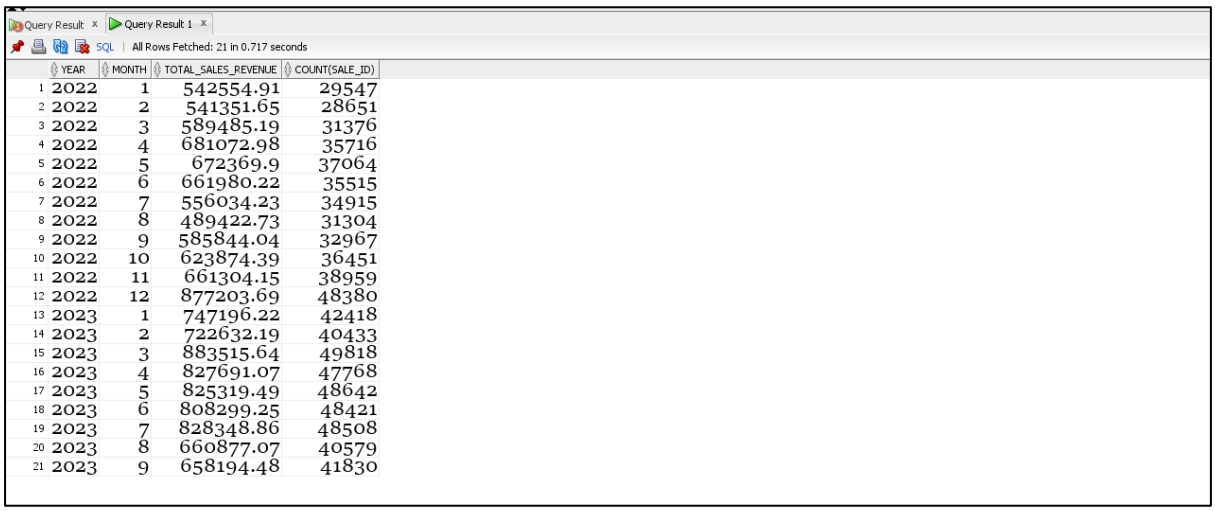
Query Result | Query Result 1

SQL | All Rows Fetched: 7 in 0.37 seconds

	QUARTER	YEAR	TOTAL_SALES_REVENUE	COUNT(SALE_ID)
1	1	2022	1673391.75	89574
2	2	2022	2015423.1	108295
3	3	2022	1631301	99186
4	4	2022	2162382.23	123790
5	1	2023	2353344.05	132669
6	2	2023	2461309.81	144831
7	3	2023	2147420.41	130917

In the quarterly sales analysis, only three quarters of 2023 are included. Upon reviewing the dataset, it was observed that some months are missing data for 2022, leading to an incomplete comparison with the sales data of 2023.
Based on the data analysis of total sales generated per month, it was observed that there are records for only 9 months in 2023.

```
SELECT  EXTRACT(YEAR FROM date_) AS YEAR,
    EXTRACT(MONTH FROM date_) AS MONTH,
    SUM(TOTAL_SALES_REVENUE) AS TOTAL_SALES_REVENUE,
    COUNT(SALE_ID)
FROM SalesSummary
GROUP BY EXTRACT(YEAR FROM date_), EXTRACT(MONTH FROM date_)
ORDER BY EXTRACT(YEAR FROM date_), EXTRACT(MONTH FROM date_);
```



Query Result | Query Result 1

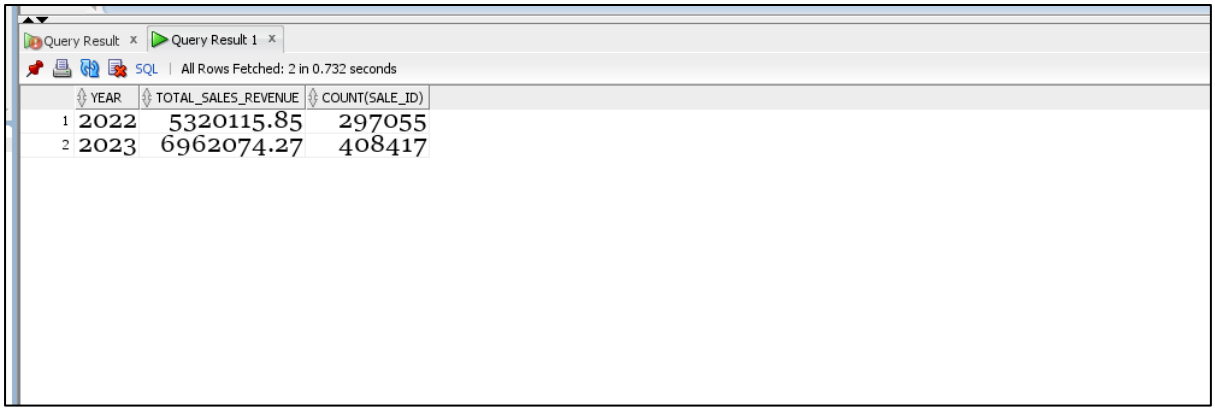
SQL | All Rows Fetched: 21 in 0.717 seconds

	YEAR	MONTH	TOTAL_SALES_REVENUE	COUNT(SALE_ID)
1	2022	1	542554.91	29547
2	2022	2	541351.65	28651
3	2022	3	589485.19	31376
4	2022	4	681072.98	35716
5	2022	5	672369.9	37064
6	2022	6	661980.22	35515
7	2022	7	556034.23	34915
8	2022	8	489422.73	31304
9	2022	9	585844.04	32967
10	2022	10	623874.39	36451
11	2022	11	661304.15	38959
12	2022	12	877203.69	48380
13	2023	1	747196.22	42418
14	2023	2	722632.19	40433
15	2023	3	883515.64	49818
16	2023	4	827691.07	47768
17	2023	5	825319.49	48642
18	2023	6	808299.25	48421
19	2023	7	828348.86	48508
20	2023	8	660877.07	40579
21	2023	9	658194.48	41830

- Based on this information, I re-checked the total sales generated per year

```
SELECT  EXTRACT(YEAR FROM date_) AS YEAR,
    SUM(TOTAL_SALES_REVENUE) AS TOTAL_SALES_REVENUE,
    COUNT(SALE_ID)
```

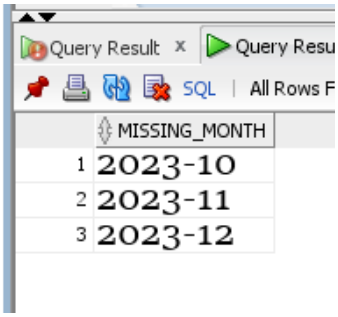
```
FROM SalesSummary
WHERE EXTRACT(MONTH FROM date_) NOT IN (10,11, 12)  -- Exclude January, February, and March
GROUP BY EXTRACT(YEAR FROM date_)
ORDER BY EXTRACT(YEAR FROM date_);
```



- Based on this information, after excluding the missing data for October to December, the total sales for 2023 still exceed those of 2022, with 2023 reaching a sales count of 408,417 compared to 2022's 297,055. the sales in 2023 grew by approximately 37.49% compared to 2022.

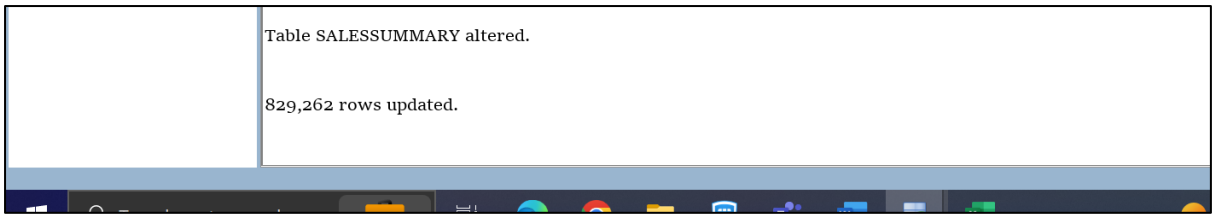
- Rechecked using Common Table Expressions (CTE) to identify which months are missing from the sales data in 2023

```
WITH Months AS (
  SELECT  ADD_MONTHS (TO_DATE('01-01-2023', 'DD-MM-YYYY'), LEVEL - 1) AS Month
  FROM dual
  CONNECT BY ADD_MONTHS(TO_DATE('01-01-2023', 'DD-MM-YYYY'), LEVEL - 1) <=
TO_DATE('31-12-2023', 'DD-MM-YYYY')
)
SELECT  TO_CHAR(Month, 'YYYY-MM') AS Missing_Month
FROM Months
WHERE NOT EXISTS (
  SELECT  1
  FROM SalesSummary
  WHERE EXTRACT(YEAR FROM date_) = 2023
  AND EXTRACT(MONTH FROM date_) = EXTRACT (MONTH FROM Month)
)
ORDER BY Month;
```



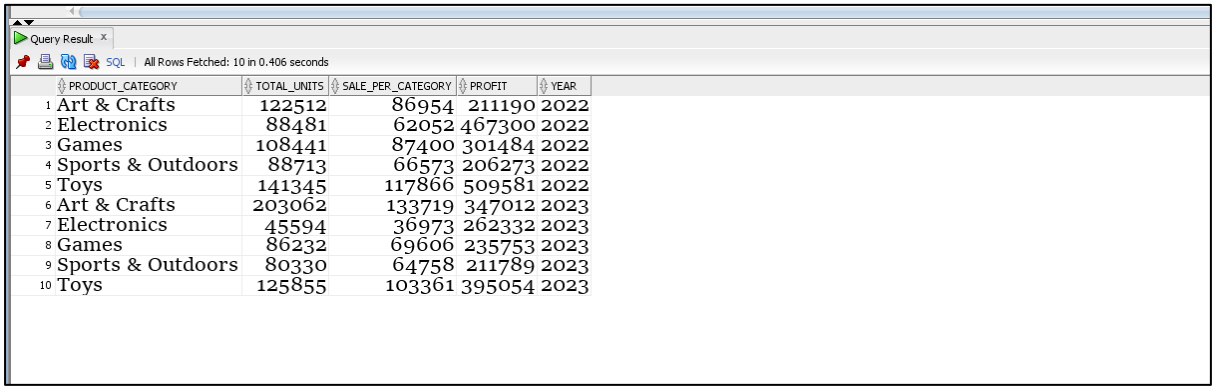

```
ALTER TABLE SalesSummary
ADD PROFIT FLOAT;
```

```
UPDATE SalesSummary
SET PROFIT = (PRODUCT_PRICE - PRODUCT_COST);
```



- Profit per product category by year

```
SELECT  Product_Category, SUM (Units) AS Total_Units, COUNT (Sale_ID) AS
Sale_per_category,
SUM (Profit) AS Profit, EXTRACT(YEAR FROM date_) AS YEAR
FROM SalesSummary
GROUP BY Product_Category, EXTRACT(YEAR FROM date_)
ORDER BY EXTRACT (YEAR FROM date_);
```



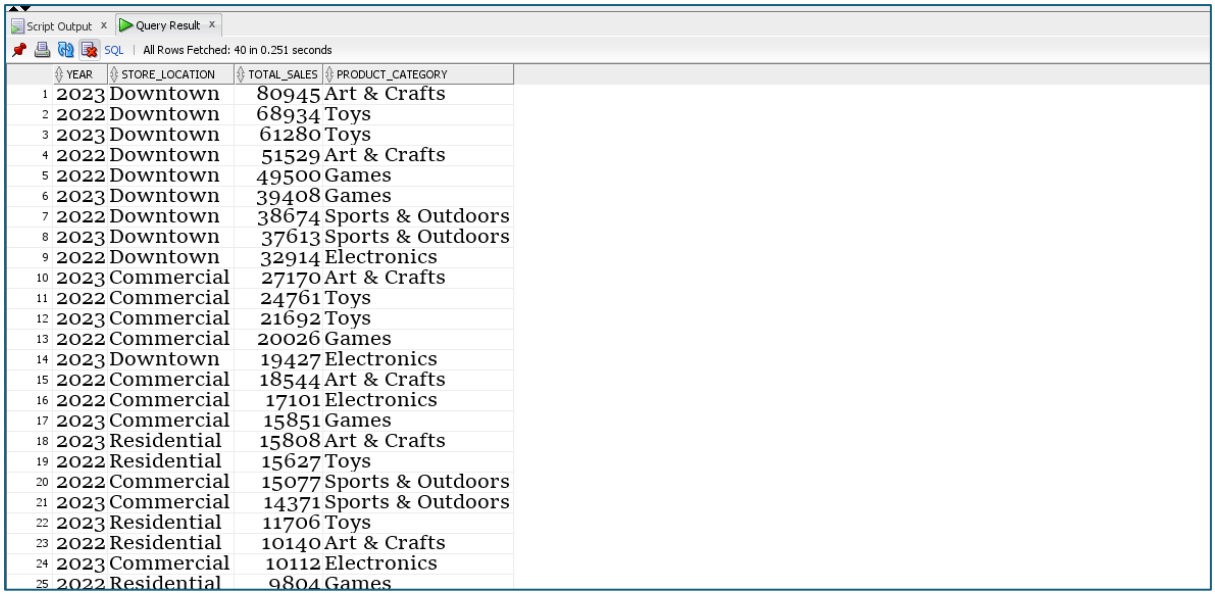
Query Result | SQL | All Rows Fetched: 10 in 0.406 seconds

	PRODUCT_CATEGORY	TOTAL_UNITS	SALE_PER_CATEGORY	PROFIT	YEAR
1	Art & Crafts	122512	86954	211190	2022
2	Electronics	88481	62052	467300	2022
3	Games	108441	87400	301484	2022
4	Sports & Outdoors	88713	66573	206273	2022
5	Toys	141345	117866	509581	2022
6	Art & Crafts	203062	133719	347012	2023
7	Electronics	45594	36973	262332	2023
8	Games	86232	69606	235753	2023
9	Sports & Outdoors	80330	64758	211789	2023
10	Toys	125855	103361	395054	2023

In 2022:
Toys had the highest sales with 141,345 units sold.
Electronics had the lowest sales with 88,481 units sold.

In 2023:
Toys continued to have the highest sales with 125,855 units sold.
Art & Crafts had the lowest sales with 122,512 units sold.

```
SELECT  EXTRACT (YEAR FROM date_) AS YEAR,
    store_location, COUNT (Sale_id) AS total_sales, Product_category
FROM salessummary
GROUP BY EXTRACT (YEAR FROM date_),store_location, Product_category
ORDER BY total_sales DESC, Product_category DESC;
```



Script Output | Query Result

SQL | All Rows Fetched: 40 in 0.251 seconds

	YEAR	STORE_LOCATION	TOTAL_SALES	PRODUCT_CATEGORY
1	2023	Downtown	80945	Art & Crafts
2	2022	Downtown	68934	Toys
3	2023	Downtown	61280	Toys
4	2022	Downtown	51529	Art & Crafts
5	2022	Downtown	49500	Games
6	2023	Downtown	39408	Games
7	2022	Downtown	38674	Sports & Outdoors
8	2023	Downtown	37613	Sports & Outdoors
9	2022	Downtown	32914	Electronics
10	2023	Commercial	27170	Art & Crafts
11	2022	Commercial	24761	Toys
12	2023	Commercial	21692	Toys
13	2022	Commercial	20026	Games
14	2023	Downtown	19427	Electronics
15	2022	Commercial	18544	Art & Crafts
16	2022	Commercial	17101	Electronics
17	2023	Commercial	15851	Games
18	2023	Residential	15808	Art & Crafts
19	2022	Residential	15627	Toys
20	2022	Commercial	15077	Sports & Outdoors
21	2023	Commercial	14371	Sports & Outdoors
22	2023	Residential	11706	Toys
23	2022	Residential	10140	Art & Crafts
24	2023	Commercial	10112	Electronics
25	2022	Residential	9804	Games

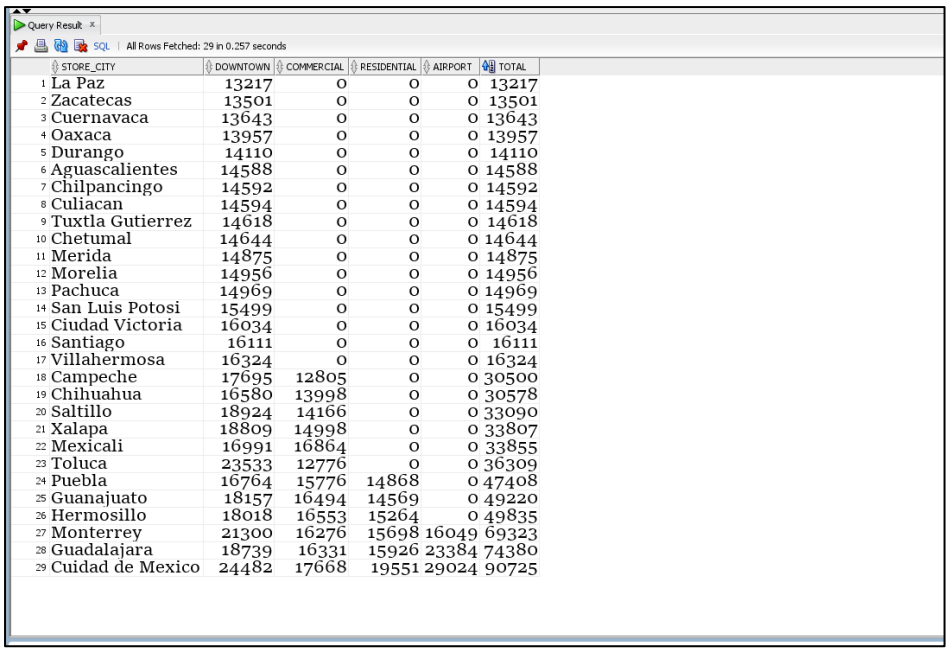
- After analyzing the sales data, I further investigated the distribution of store IDs across different regions.

```
SELECT  STORE_LOCATION, COUNT(DISTINCT STORE_ID) AS
NUMBER_OF_STORE_ASSIGNED FROM A_STOR
GROUP BY STORE_LOCATION
```

	STORE_LOCATION	NUMBER_OF_STORE_ASSIGNED
1	Residential	6
2	Commercial	12
3	Downtown	29
4	Airport	3

- I examined the distribution of stores across different regions like Downtown, Commercial, Residential, Airport

```
SELECT
  Store_City, Downtown, Commercial, Residential, Airport,
  (Downtown + Commercial + Residential + Airport) AS Total
FROM (
  SELECT  Store_City, Store_Location
  FROM SalesSummary)
PIVOT (
  COUNT(Store_Location)
  FOR Store_Location IN ('Downtown' AS Downtown, 'Commercial' AS Commercial,
'Residential' AS Residential, 'Airport' AS Airport))
ORDER BY Store_City;
```



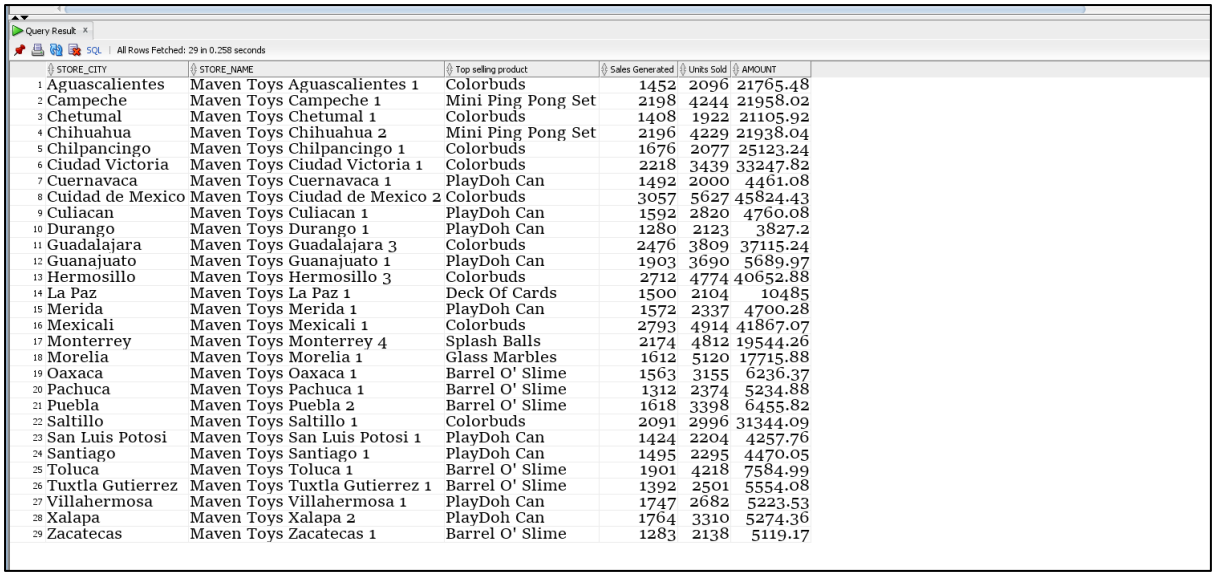
Query Result

All Rows Fetched: 29 in 0.257 seconds

	STORE_CITY	DOWNTOWN	COMMERCIAL	RESIDENTIAL	AIRPORT	TOTAL
1	La Paz	13217	0	0	0	13217
2	Zacatecas	13501	0	0	0	13501
3	Cuernavaca	13643	0	0	0	13643
4	Oaxaca	13957	0	0	0	13957
5	Durango	14110	0	0	0	14110
6	Aguascalientes	14588	0	0	0	14588
7	Chilpancingo	14592	0	0	0	14592
8	Culiacan	14594	0	0	0	14594
9	Tuxtla Gutierrez	14618	0	0	0	14618
10	Chetumal	14644	0	0	0	14644
11	Merida	14875	0	0	0	14875
12	Morelia	14956	0	0	0	14956
13	Pachuca	14969	0	0	0	14969
14	San Luis Potosi	15499	0	0	0	15499
15	Ciudad Victoria	16034	0	0	0	16034
16	Santiago	16111	0	0	0	16111
17	Villahermosa	16324	0	0	0	16324
18	Campeche	17695	12805	0	0	30500
19	Chihuahua	16580	13998	0	0	30578
20	Saltillo	18924	14166	0	0	33090
21	Xalapa	18809	14998	0	0	33807
22	Mexicali	16991	16864	0	0	33855
23	Toluca	23533	12776	0	0	36309
24	Puebla	16764	15776	14868	0	47408
25	Guanajuato	18157	16494	14569	0	49220
26	Hermosillo	18018	16553	15264	0	49835
27	Monterrey	21300	16276	15698	16049	69323
28	Guadalajara	18739	16331	15926	23384	74380
29	Cuidad de Mexico	24482	17668	19551	29024	90725

- The top-selling products in each store city are determined based on the number of units sold.

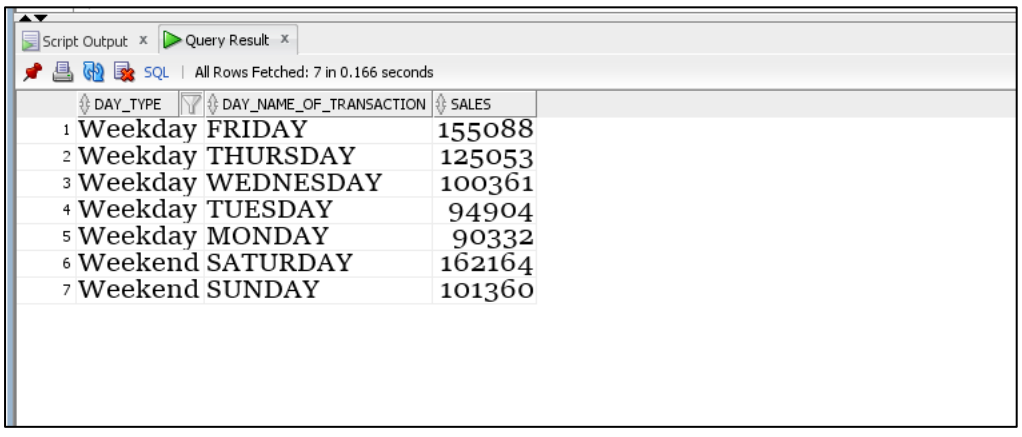
```
WITH Top_selling_products AS (
  SELECT
    Store_City,
    Store_Name,
    Product_Name,
    COUNT(Sale_ID) AS sales,
    SUM(Units) AS units,
    SUM(PRODUCT_PRICE) P,
    ROW_NUMBER() OVER (PARTITION BY Store_City ORDER BY SUM(Units) DESC) AS
Products_ranked
  FROM
    SalesSummary
  GROUP BY
    Store_City,
    Product_Name,
    Store_Name
)
SELECT
  Store_City,
  Store_Name,
  Product_Name AS "Top selling product",
  sales AS "Sales Generated",
  units AS "Units Sold", P AS Amount
FROM
  Top_selling_products
WHERE
  Products_ranked = 1;
```



	STORE_CITY	STORE_NAME	Top selling product	Sales Generated	Units Sold	AMOUNT
1	Aguascalientes	Maven Toys Aguascalientes 1	Colorbuds	1452	2096	21765.48
2	Campeche	Maven Toys Campeche 1	Mini Ping Pong Set	2198	4244	21958.02
3	Chetumal	Maven Toys Chetumal 1	Colorbuds	1408	1922	21105.92
4	Chihuahua	Maven Toys Chihuahua 2	Mini Ping Pong Set	2196	4229	21938.04
5	Chilpancingo	Maven Toys Chilpancingo 1	Colorbuds	1676	2077	25123.24
6	Ciudad Victoria	Maven Toys Ciudad Victoria 1	Colorbuds	2218	3439	33247.82
7	Cuernavaca	Maven Toys Cuernavaca 1	PlayDoh Can	1492	2000	4461.08
8	Cuidad de Mexico	Maven Toys Ciudad de Mexico 2	Colorbuds	3057	5627	45824.43
9	Culiacan	Maven Toys Culiacan 1	PlayDoh Can	1592	2820	4760.08
10	Durango	Maven Toys Durango 1	PlayDoh Can	1280	2123	3827.2
11	Guadalajara	Maven Toys Guadalajara 3	Colorbuds	2476	3809	37115.24
12	Guanajuato	Maven Toys Guanajuato 1	PlayDoh Can	1903	3690	5689.97
13	Hermosillo	Maven Toys Hermosillo 3	Colorbuds	2712	4774	40652.88
14	La Paz	Maven Toys La Paz 1	Deck Of Cards	1500	2104	10485
15	Merida	Maven Toys Merida 1	PlayDoh Can	1572	2337	4700.28
16	Mexicali	Maven Toys Mexicali 1	Colorbuds	2793	4914	41867.07
17	Monterrey	Maven Toys Monterrey 4	Splash Balls	2174	4812	19544.26
18	Morelia	Maven Toys Morelia 1	Glass Marbles	1612	5120	17715.88
19	Oaxaca	Maven Toys Oaxaca 1	Barrel O' Slime	1563	3155	6236.37
20	Pachuca	Maven Toys Pachuca 1	Barrel O' Slime	1312	2374	5234.88
21	Puebla	Maven Toys Puebla 2	Barrel O' Slime	1618	3398	6455.82
22	Saltillo	Maven Toys Saltillo 1	Colorbuds	2091	2996	31344.09
23	San Luis Potosi	Maven Toys San Luis Potosi 1	PlayDoh Can	1424	2204	4257.76
24	Santiago	Maven Toys Santiago 1	PlayDoh Can	1495	2295	4470.05
25	Toluca	Maven Toys Toluca 1	Barrel O' Slime	1901	4218	7584.99
26	Tuxtla Gutierrez	Maven Toys Tuxtla Gutierrez 1	Barrel O' Slime	1392	2501	5554.08
27	Villahermosa	Maven Toys Villahermosa 1	PlayDoh Can	1747	2682	5223.53
28	Xalapa	Maven Toys Xalapa 2	PlayDoh Can	1764	3310	5274.36
29	Zacatecas	Maven Toys Zacatecas 1	Barrel O' Slime	1283	2138	5119.17

- Next, I analyzed the sales data to differentiate between weekday and weekend sales.

```
WITH new_name AS (
  SELECT
    CASE WHEN Day_Name_Of_Transaction IN ('Saturday', 'Sunday') THEN 'Weekend'
      ELSE 'Weekday'
    END AS Day_Type, Day_Name_Of_Transaction,  COUNT(Sale_ID) AS Sales
  FROM SalesSummary
  GROUP BY Day_Name_Of_Transaction)
SELECT  *  FROM new_name
ORDER BY Day_Type, Sales DESC;
```



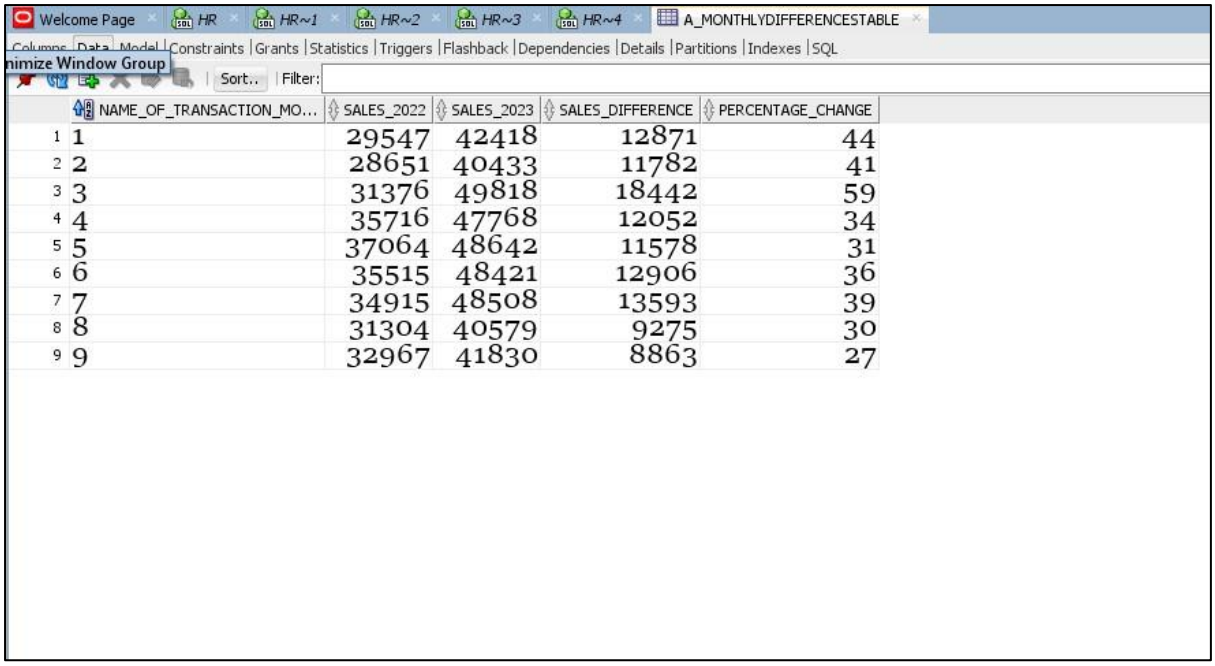
Script Output | Query Result | SQL | All Rows Fetched: 7 in 0.166 seconds

	DAY_TYPE	DAY_NAME_OF_TRANSACTION	SALES
1	Weekday	FRIDAY	155088
2	Weekday	THURSDAY	125053
3	Weekday	WEDNESDAY	100361
4	Weekday	TUESDAY	94904
5	Weekday	MONDAY	90332
6	Weekend	SATURDAY	162164
7	Weekend	SUNDAY	101360

- Predicted sales and percentage change for October, November, and December 2023 are as follows:

```
CREATE TABLE A_MonthlyDifferencesTable (
  Name_Of_Transaction_Month VARCHAR(20),
  Sales_2022 INT,
  Sales_2023 INT,
  Sales_Difference INT,
  Percentage_Change FLOAT);


INSERT INTO A_MonthlyDifferencesTable (Name_Of_Transaction_Month, Sales_2022,
Sales_2023, Sales_Difference, Percentage_Change)
WITH MonthlySales AS (
  SELECT
    EXTRACT(MONTH FROM Date_) AS Name_Of_Transaction_Month,
    EXTRACT(YEAR FROM Date_) AS Year_Of_Transaction,
    COUNT(Sale_ID) AS Sales
  FROM SalesSummary
  WHERE (EXTRACT(YEAR FROM Date_) = 2022 AND EXTRACT(MONTH FROM Date_) <= 12)
   OR (EXTRACT(YEAR FROM Date_) = 2023 AND EXTRACT(MONTH FROM Date_) <= 12)
  GROUP BY EXTRACT(MONTH FROM Date_), EXTRACT(YEAR FROM Date_)
),
MonthlyDifferences AS (
  SELECT
    M1.Name_Of_Transaction_Month,
    M1.Sales AS Sales_2022,
    M2.Sales AS Sales_2023,
    M2.Sales - M1.Sales AS Sales_Difference,
    ROUND((M2.Sales - M1.Sales) / CAST(M1.Sales AS FLOAT) * 100, 0) AS Percentage_Change
  FROM MonthlySales M1
  JOIN MonthlySales M2 ON M1.Name_Of_Transaction_Month =
M2.Name_Of_Transaction_Month
                AND M1.Year_Of_Transaction = 2022
                AND M2.Year_Of_Transaction = 2023
)
SELECT  * FROM MonthlyDifferences;
```

```
Table A_MONTHLYDIFFERENCESTABLE created.

9 rows inserted.
```



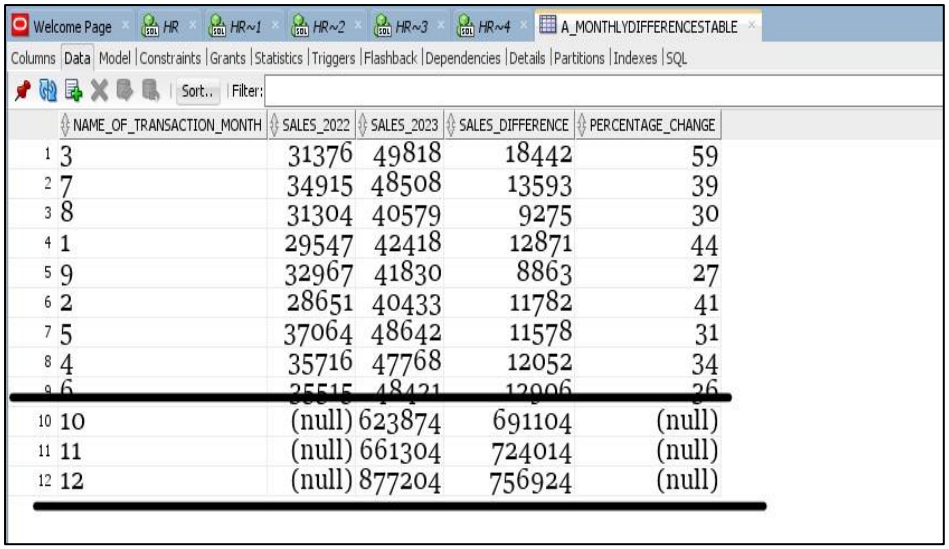
	NAME_OF_TRANSACTION_MO...	SALES_2022	SALES_2023	SALES_DIFFERENCE	PERCENTAGE_CHANGE
1	1	29547	42418	12871	44
2	2	28651	40433	11782	41
3	3	31376	49818	18442	59
4	4	35716	47768	12052	34
5	5	37064	48642	11578	31
6	6	35515	48421	12906	36
7	7	34915	48508	13593	39
8	8	31304	40579	9275	30
9	9	32967	41830	8863	27

-

-- Insert forecasted sales for October, November, and December 2023 into the MonthlyDifferences table

```
INSERT INTO A_MonthlyDifferencesTable (Name_Of_Transaction_Month, Sales_2022, Sales_2023)
WITH MonthlySales AS (
  -- Calculate total sales revenue for each month in 2022 and 2023
  SELECT
    EXTRACT(MONTH FROM Date_) AS Month,
    EXTRACT(YEAR FROM Date_) AS Year,
    COUNT(CASE WHEN EXTRACT(YEAR FROM Date_) = 2022 THEN Sale_ID END) AS Sales_2022,
    COUNT(CASE WHEN EXTRACT(YEAR FROM Date_) = 2023 THEN Sale_ID END) AS Sales_2023
  FROM SalesSummary
  WHERE EXTRACT(YEAR FROM Date_) IN (2022, 2023)
  GROUP BY EXTRACT(YEAR FROM Date_), EXTRACT(MONTH FROM Date_)
),
Forecast AS (
  -- Generate sales forecasts for October, November, and December 2023
  SELECT
    10 AS Name_Of_Transaction_Month,
   (SELECT  COUNT(Sale_ID)
    FROM  SalesSummary
    WHERE EXTRACT(YEAR FROM Date_) = 2022 AND EXTRACT(MONTH FROM Date_) = 10 ) AS Sales_2022,
    (Sales_2023 * 1.05) AS Sales_2023,-- Example: increase sales by 5% for simplicity
```

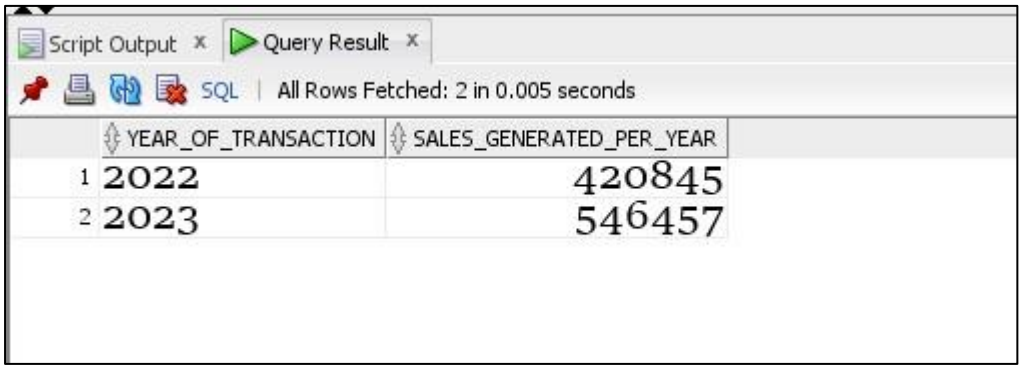
```
  FROM MonthlySales
  WHERE Year = 2023 AND Month = 9
  UNION ALL
  SELECT
    11 AS Name_Of_Transaction_Month,
    (SELECT  COUNT(Sale_ID)
   FROM  SalesSummary
   WHERE EXTRACT(YEAR FROM Date_) = 2022 AND EXTRACT(MONTH FROM Date_) = 11 ) AS
Sales_2022,
    (Sales_2023 * 1.1) AS Sales_2023  -- Example: Increase sales by 10% for simplicity
  FROM MonthlySales
  WHERE Year = 2023 AND Month = 9
  UNION ALL
  SELECT
    12 AS Name_Of_Transaction_Month,
    (SELECT  COUNT(Sale_ID)
   FROM  SalesSummary
   WHERE EXTRACT(YEAR FROM Date_) = 2022 AND EXTRACT(MONTH FROM Date_) = 12 )
AS Sales_2022,
    (Sales_2023 * 1.15) AS Sales_2023  -- Example: Increase sales by 15% for simplicity
  FROM MonthlySales
  WHERE Year = 2023 AND Month = 9

)
SELECT  * FROM Forecast;
```



	NAME_OF_TRANSACTION_MONTH	SALES_2022	SALES_2023	SALES_DIFFERENCE	PERCENTAGE_CHANGE
1	3	31376	49818	18442	59
2	7	34915	48508	13593	39
3	8	31304	40579	9275	30
4	1	29547	42418	12871	44
5	9	32967	41830	8863	27
6	2	28651	40433	11782	41
7	5	37064	48642	11578	31
8	4	35716	47768	12052	34
9	6	35515	48421	12906	36
10	10	(null)	623874	691104	(null)
11	11	(null)	661304	724014	(null)
12	12	(null)	877204	756924	(null)

Total sales generated per year, including predicted amounts for October to December 2023

```
SELECT  '2022' AS Year_Of_Transaction, SUM (Sales_2022) AS Sales_Generated_per_year
FROM A_MonthlyDifferencesTable
UNION ALL
SELECT  '2023' AS Year_Of_Transaction, SUM (Sales_2023) AS Sales_Generated_per_year
FROM A_MonthlyDifferencesTable;
```



Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.005 seconds

	YEAR_OF_TRANSACTION	SALES_GENERATED_PER_YEAR
1	2022	420845
2	2023	546457