SQL PROJECT ON

GROCERY SALES

PROJECT INTRODUCTION

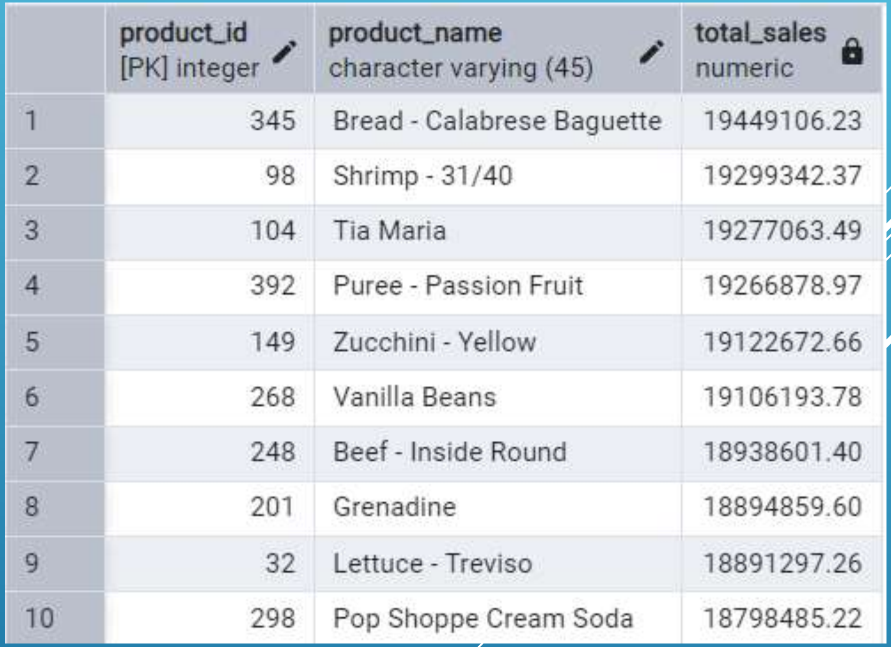
IN THIS PROJECT, I HAVE USED SQL QUERIES TO SOLVE VARIOUS QUESTIONS RELATED TO GROCERY SALES DATA.

QUERIES

1. IDENTIFY THE TOP 10 BEST-SELLING PRODUCTS, BASED ON THEIR SALES
2. FIND OUT HOW EACH PRODUCT CATEGORY PERFORMS IN TERMS OF SALES
3. HOW MANY EMPLOYEES DO WE CURRENTLY HAVE?
4. WE WANT TO REWARD OUR BEST EMPLOYEES . FIND OUT HOW MANY CUSTOMERS EACH EMPLOYEE SERVED.
5. WE WANT TO KNOW, HOW SALES ARE EVOLVING ACROSS DIFFERENT CITIES. DISPLAY THE TOTAL SALES FOR EACH CITY AND SORT THEM BY PERFORMANCE.
6. WHAT IS THE AVERAGE ORDER VALUE IN EACH CITY?
7. WE WANT A QUICK OVERVIEW TO SEE WHETHER THE COMPANY IS GROWING OR STAGNATING. COMPARE THE MONTHLY SALES.
8. WHICH DAY OF THE WEEK HAS THE HIGHEST SALES? WE WANT TO BASE OUR MARKETING CAMPAIGN ON THAT.
9. DETERMINE THE DISTRIBUTION OF SALES BY HOUR OF THE DAY.

1. IDENTIFY THE TOP 10 BEST-SELLING PRODUCTS, BASED ON THEIR SALES

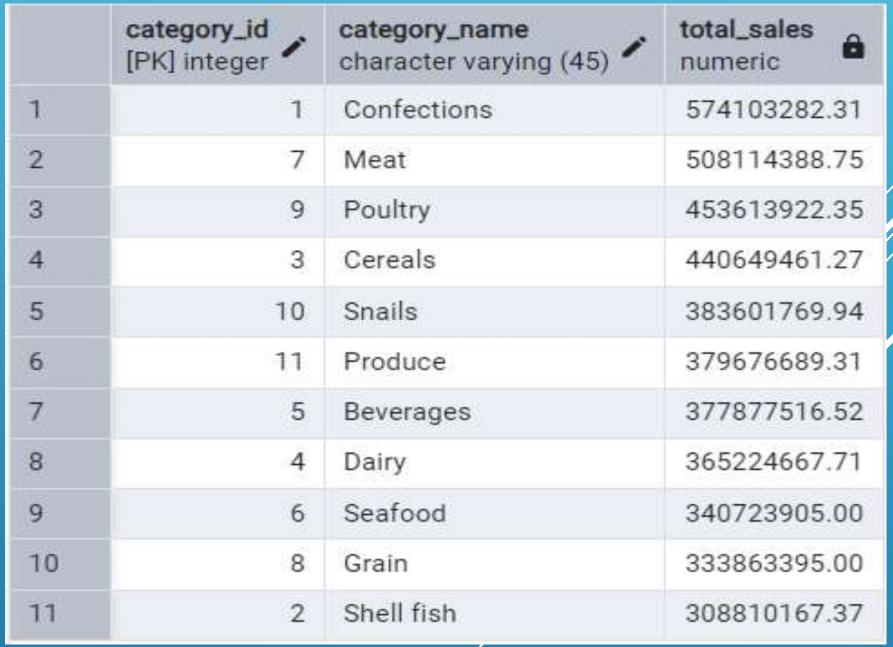
```
SELECT product.product_id, product.product_name,
ROUND(SUM (sale.quantity * product.price),2) AS total_sales
FROM product
JOIN sale
ON product.product_id = sale.product_id
GROUP BY product.product_id, product.product_name
ORDER BY ROUND(SUM (sale.quantity * product.price),2) DESC
LIMIT 10;
```



	product_id [PK] integer	product_name character varying (45)	total_sales numeric
1	345	Bread - Calabrese Baguette	19449106.23
2	98	Shrimp - 31/40	19299342.37
3	104	Tia Maria	19277063.49
4	392	Puree - Passion Fruit	19266878.97
5	149	Zucchini - Yellow	19122672.66
6	268	Vanilla Beans	19106193.78
7	248	Beef - Inside Round	18938601.40
8	201	Grenadine	18894859.60
9	32	Lettuce - Treviso	18891297.26
10	298	Pop Shoppe Cream Soda	18798485.22

2. FIND OUT HOW EACH PRODUCT CATEGORY PERFORMS IN TERMS OF SALES

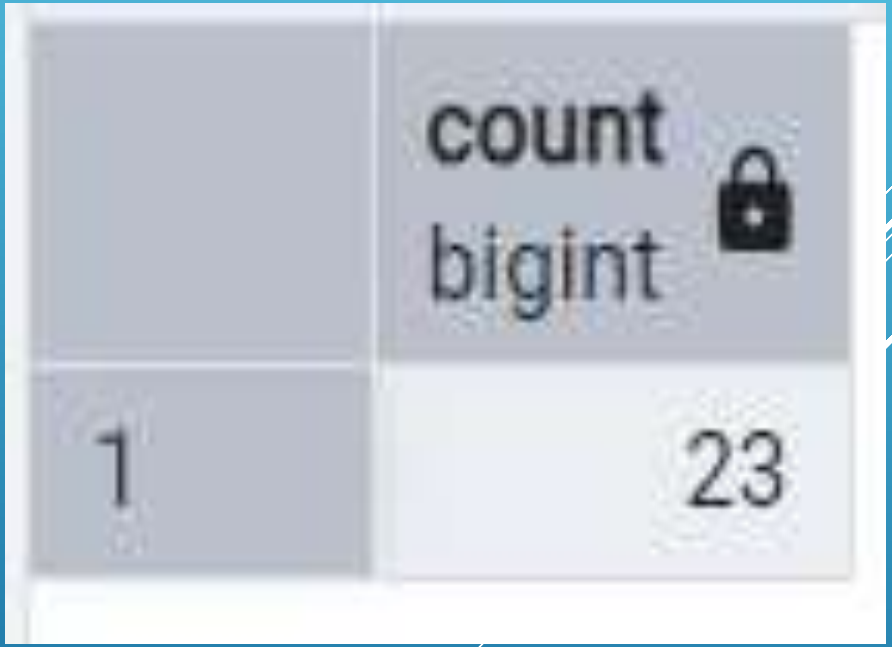
```
SELECT category.category_id, category.category_name,
ROUND(SUM(sale.quantity * product.price),2) AS total_sales
FROM sale
JOIN product
ON sale.product_id = product.product_id
JOIN category
ON product.category_id = category.category_id
GROUP BY category.category_id, category.category_name
ORDER BY total_sales DESC;
```



	category_id [PK] integer	category_name character varying (45)	total_sales numeric
1	1	Confections	574103282.31
2	7	Meat	508114388.75
3	9	Poultry	453613922.35
4	3	Cereals	440649461.27
5	10	Snails	383601769.94
6	11	Produce	379676689.31
7	5	Beverages	377877516.52
8	4	Dairy	365224667.71
9	6	Seafood	340723905.00
10	8	Grain	333863395.00
11	2	Shell fish	308810167.37

3. HOW MANY EMPLOYEES DO WE CURRENTLY HAVE?

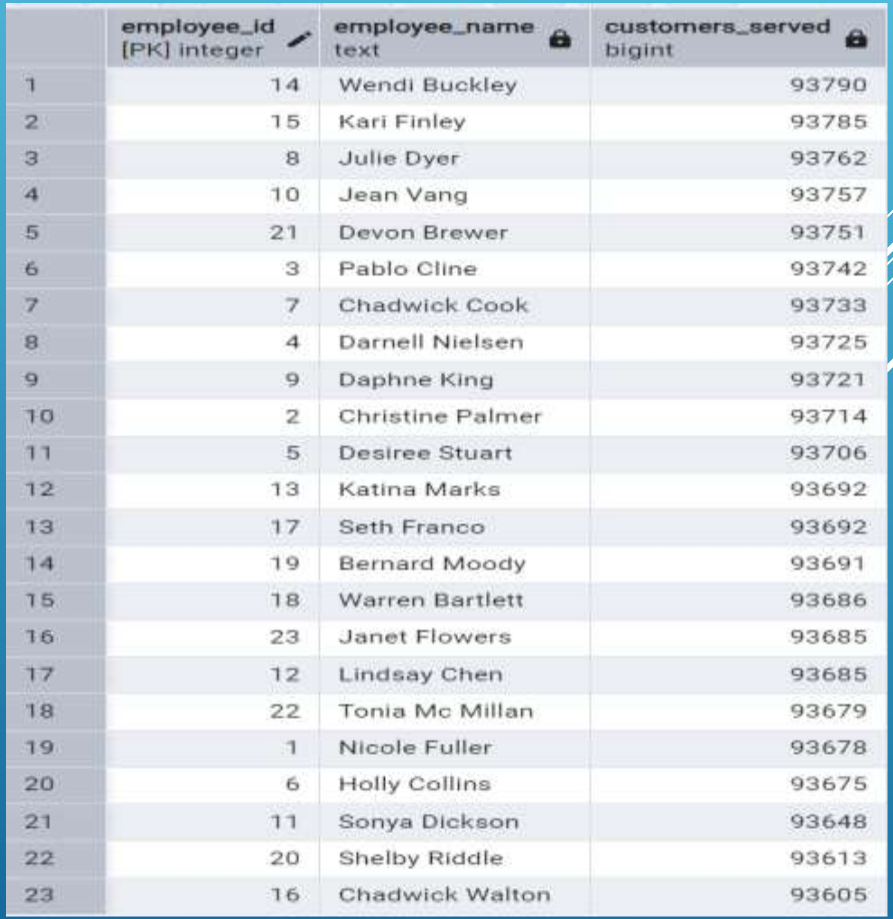
```sql
SELECT COUNT(*) FROM employee;
```



	count bigint
1	23

4. FIND OUT HOW MANY CUSTOMERS EACH EMPLOYEE SERVED

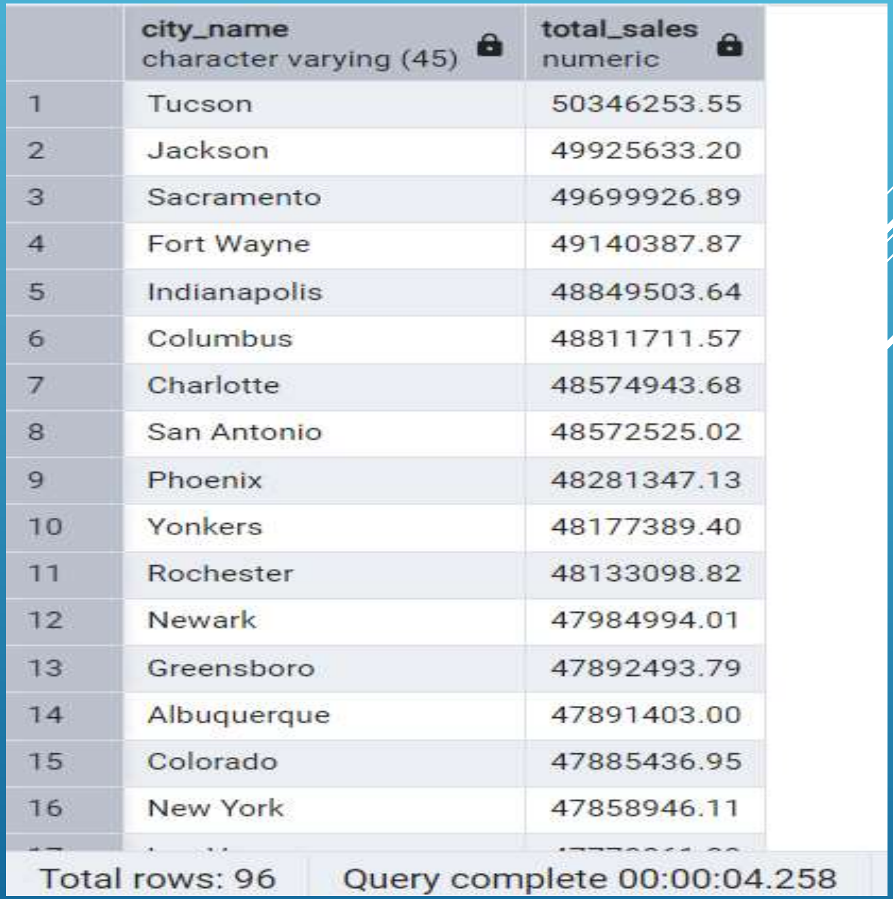
```
SELECT employee.employee_id,
employee.first_name || ' ' || employee.last_name AS employee_name,
COUNT(DISTINCT sale.customer_id) AS customers_served
FROM employee
JOIN sale
ON employee.employee_id = sale.sales_person_id
GROUP BY employee.employee_id, employee.first_name, employee.last_name
ORDER BY customers_served DESC;
```



	employee_id [PK] integer	employee_name text	customers_served bigint
1	14	Wendi Buckley	93790
2	15	Kari Finley	93785
3	8	Julie Dyer	93762
4	10	Jean Vang	93757
5	21	Devon Brewer	93751
6	3	Pablo Cline	93742
7	7	Chadwick Cook	93733
8	4	Darnell Nielsen	93725
9	9	Daphne King	93721
10	2	Christine Palmer	93714
11	5	Desiree Stuart	93706
12	13	Katina Marks	93692
13	17	Seth Franco	93692
14	19	Bernard Moody	93691
15	18	Warren Bartlett	93686
16	23	Janet Flowers	93685
17	12	Lindsay Chen	93685
18	22	Tonia Mc Millan	93679
19	1	Nicole Fuller	93678
20	6	Holly Collins	93675
21	11	Sonya Dickson	93648
22	20	Shelby Riddle	93613
23	16	Chadwick Walton	93605

5. DISPLAY THE TOTAL SALES FOR EACH CITY AND SORT THEM BY PERFORMANCE

```
SELECT city.city_name, ROUND(SUM(sale.quantity * product.price),2) AS total_sales
FROM sale
JOIN product
ON product.product_id = sale.product_id
JOIN customer
ON customer.customer_id = sale.customer_id
JOIN city
ON city.city_id = customer.city_id
GROUP BY city_name
ORDER BY total_sales DESC;
```

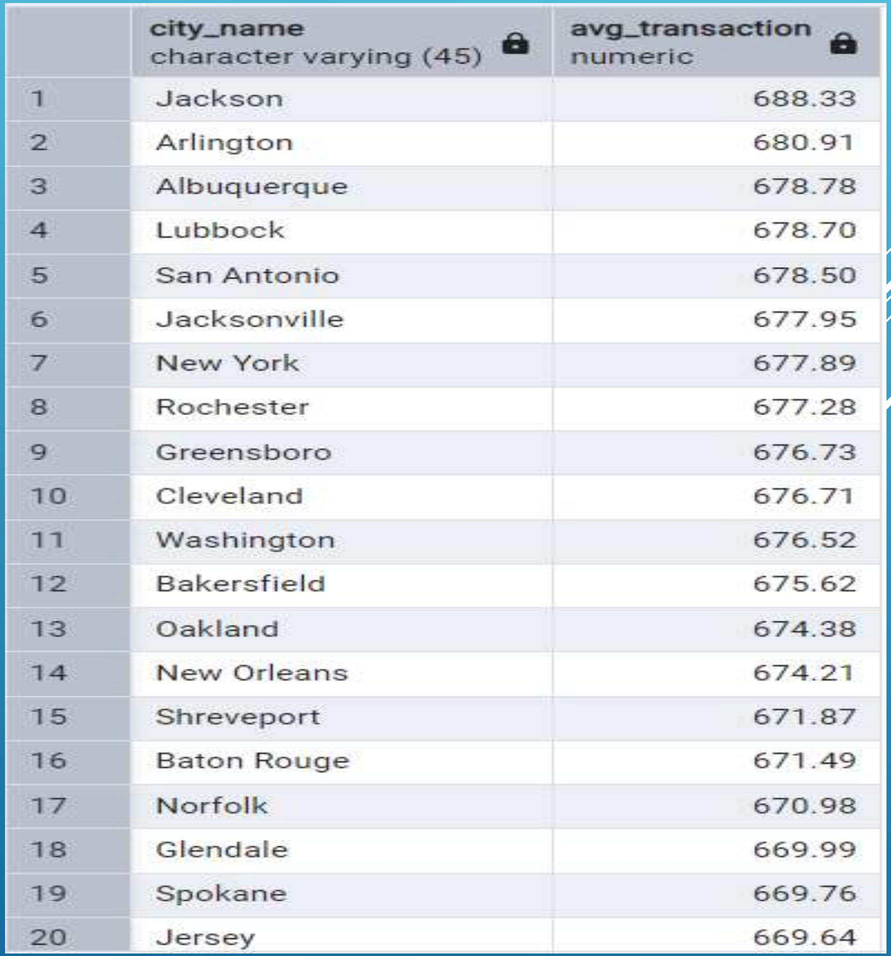


	city_name character varying (45)	total_sales numeric
1	Tucson	50346253.55
2	Jackson	49925633.20
3	Sacramento	49699926.89
4	Fort Wayne	49140387.87
5	Indianapolis	48849503.64
6	Columbus	48811711.57
7	Charlotte	48574943.68
8	San Antonio	48572525.02
9	Phoenix	48281347.13
10	Yonkers	48177389.40
11	Rochester	48133098.82
12	Newark	47984994.01
13	Greensboro	47892493.79
14	Albuquerque	47891403.00
15	Colorado	47885436.95
16	New York	47858946.11

Total rows: 96 Query complete 00:00:04.258

6. WHAT IS THE AVERAGE ORDER VALUE IN EACH CITY?

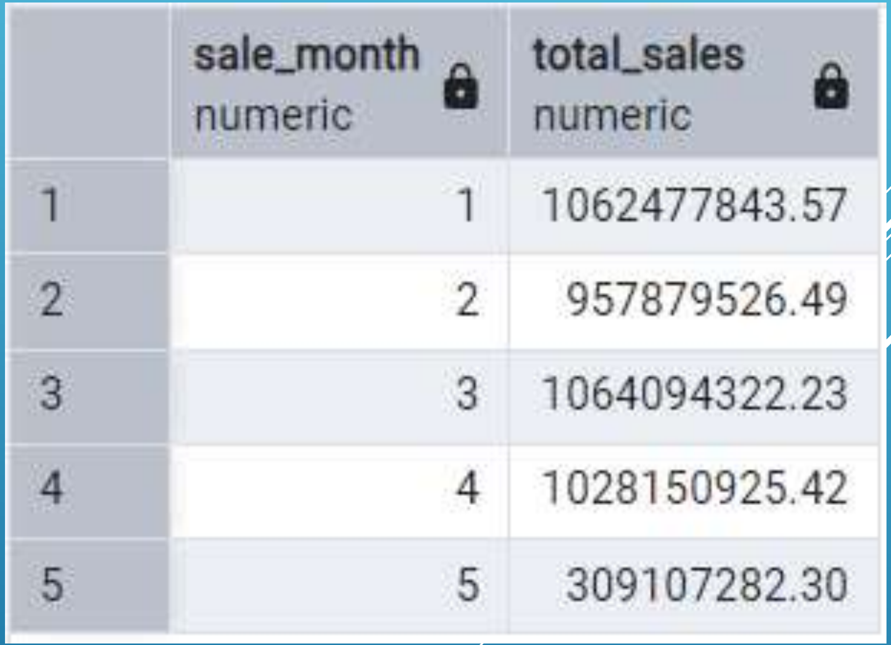
```
SELECT city.city_name,
ROUND(SUM(sale.quantity * product.price)/COUNT(DISTINCT
sale.sales_id),2) AS avg_transaction
FROM sale
JOIN product
ON product.product_id = sale.product_id
JOIN customer
ON customer.customer_id = sale.customer_id
JOIN city
ON city.city_id = customer.city_id
GROUP BY city_name
ORDER BY avg_transaction DESC;
```



	city_name character varying (45)	avg_transaction numeric
1	Jackson	688.33
2	Arlington	680.91
3	Albuquerque	678.78
4	Lubbock	678.70
5	San Antonio	678.50
6	Jacksonville	677.95
7	New York	677.89
8	Rochester	677.28
9	Greensboro	676.73
10	Cleveland	676.71
11	Washington	676.52
12	Bakersfield	675.62
13	Oakland	674.38
14	New Orleans	674.21
15	Shreveport	671.87
16	Baton Rouge	671.49
17	Norfolk	670.98
18	Glendale	669.99
19	Spokane	669.76
20	Jersey	669.64

7. COMPARE THE MONTHLY SALES

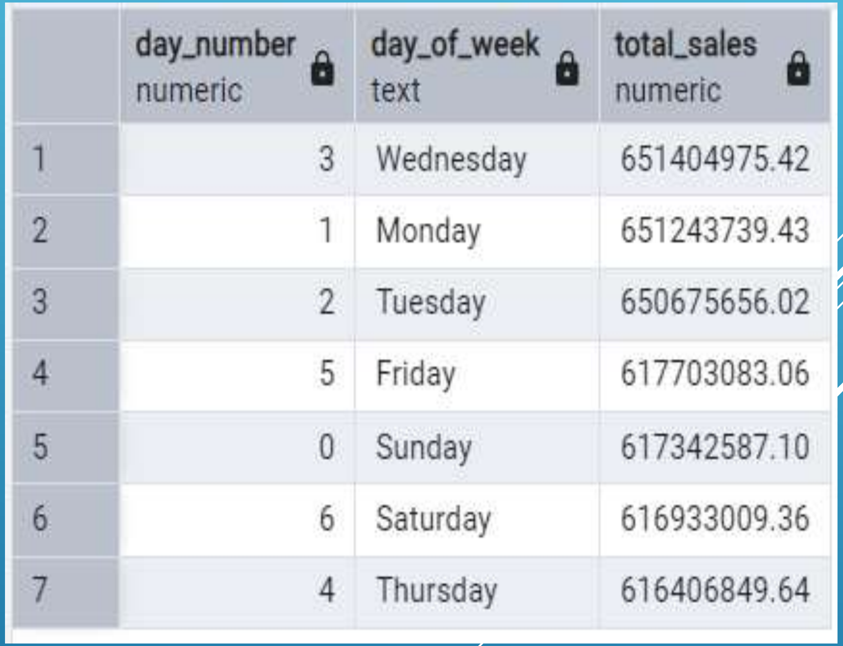
```
SELECT
EXTRACT (MONTH FROM sale.sales_date) AS sale_month,
ROUND(SUM(sale.quantity * product.price),2) AS total_sales
FROM sale
JOIN product
ON sale.product_id = product.product_id
WHERE sale.sales_date IS NOT NULL
GROUP BY sale_month
ORDER BY sale_month;
```



	sale_month numeric	total_sales numeric
1	1	1062477843.57
2	2	957879526.49
3	3	1064094322.23
4	4	1028150925.42
5	5	309107282.30

8. WHICH DAY OF THE WEEK HAS THE HIGHEST SALES?

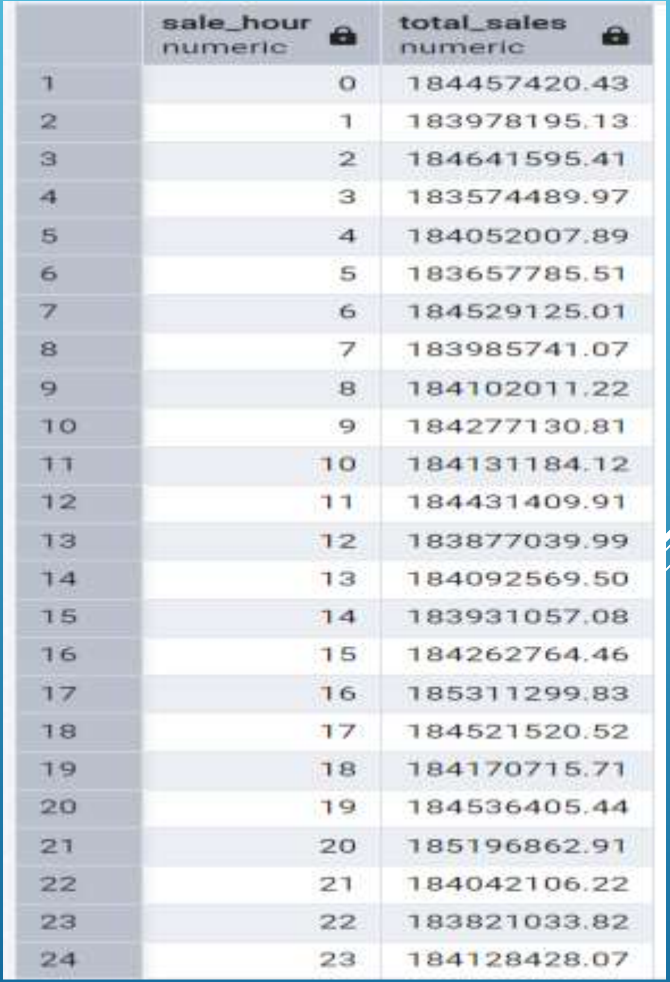
```
SELECT EXTRACT(DOW FROM sale.sales_date) AS day_number,
TO_CHAR(sale.sales_date, 'Day') AS day_of_week,
ROUND(SUM(sale.quantity * product.price),2) AS total_sales
FROM sale
JOIN product
ON sale.product_id = product.product_id
WHERE sale.sales_date IS NOT NULL
GROUP BY day_number, day_of_week
ORDER BY total_sales DESC;
```



	day_number numeric	day_of_week text	total_sales numeric
1	3	Wednesday	651404975.42
2	1	Monday	651243739.43
3	2	Tuesday	650675656.02
4	5	Friday	617703083.06
5	0	Sunday	617342587.10
6	6	Saturday	616933009.36
7	4	Thursday	616406849.64

9. DETERMINE THE DISTRIBUTION OF SALES BY HOUR OF THE DAY

```
SELECT
EXTRACT (HOUR FROM sale.sales_date) AS sale_hour,
ROUND(SUM(sale.quantity * product.price),2) AS total_sales
FROM sale
JOIN product
ON sale.product_id = product.product_id
WHERE sale.sales_date IS NOT NULL
GROUP BY sale_hour
ORDER BY sale_hour;
```



	sale_hour numeric	total_sales numeric
1	0	184457420.43
2	1	183978195.13
3	2	184641595.41
4	3	183574489.97
5	4	184052007.89
6	5	183657785.51
7	6	184529125.01
8	7	183985741.07
9	8	184102011.22
10	9	184277130.81
11	10	184131184.12
12	11	184431409.91
13	12	183877039.99
14	13	184092569.50
15	14	183931057.08
16	15	184262764.46
17	16	185311299.83
18	17	184521520.52
19	18	184170715.71
20	19	184536405.44
21	20	185196862.91
22	21	184042106.22
23	22	183821033.82
24	23	184128428.07