



INNOMATICS[®]
RESEARCH LABS

INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

SQL PROJECT ON GROCERY STORE MANAGEMENT

GROCERY STORE MANAGEMENT SYSTEM

ABOUT US

G.T.SIDDHARDHA

BATCH - 381

**BACKGROUND-
BTECH(CSE-AI)**

**M. CHAITANYA
KRISHNA REDDY**

BATCH-381

**BACKGROUND -
BTECH(ECE)**

SK. SIDDIQ

BATCH-381

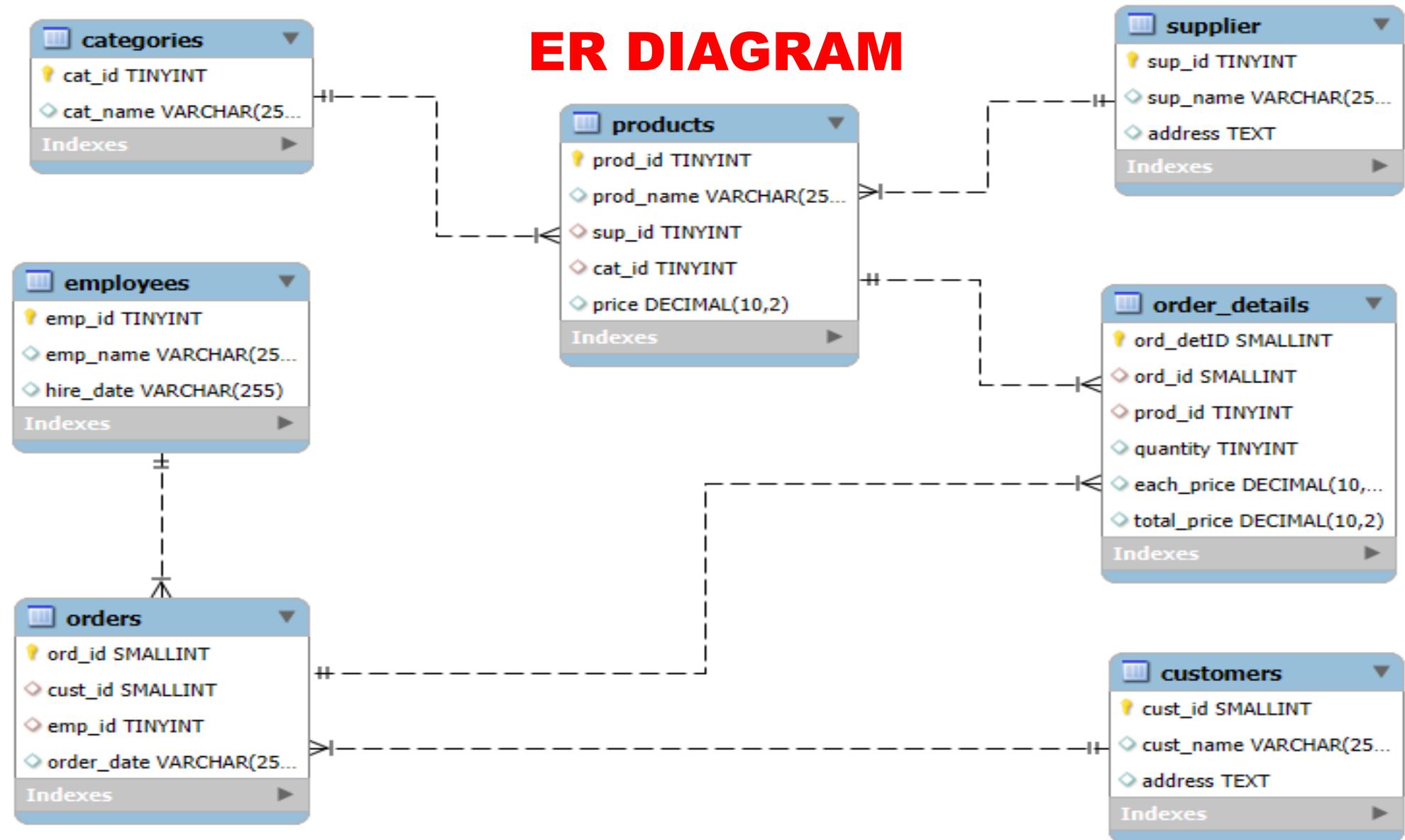
**BACKGROUND-
BTECH(CSE)**

INTRODUCTION

The Grocery Store Management System is a SQL-based project that manages products, customers, suppliers, employees, and orders through a relational database. It enables efficient tracking of sales, inventory, and operations while allowing data analysis for better business decisions using SQL queries and insights.

ENTITY	DESCRIPTION
SUPPLIERS	External vendors who provide grocery items to the store.
CATEGORIES	Classification of products into types such as Beverages, Snacks, Dairy, etc.
EMPLOYEES	Staff members responsible for handling customer orders and store operations.
CUSTOMERS	Individuals who purchase products and place orders at the grocery store.
PRODUCTS	Grocery items available for sale, each linked to a category and a supplier.
ORDERS	Purchase transactions initiated by customers and processed by employees.
ORDER_DETAILS	Detailed records of products within each order, including quantity and pricing.

ER DIAGRAM



EXISTING RELATIONS

- ✓ **One-to-Many from supplier to products**
- ✓ **One-to-Many from categories to products**
- ✓ **One-to-Many from products to order_details**
- ✓ **One-to-Many from orders to order_details**
- ✓ **One-to-Many from customers to orders**
- ✓ **One-to-Many from employees to orders**

DATABASE SCHEMA CREATION

TABLE NAME	COLUMNS (Primary and Foreign Keys Included)
SUPPLIER	SUP_ID (PK), SUP_NAME , ADDRESS
CATEGORIES	CAT_ID (PK), CAT_NAME
EMPLOYEES	EMP_ID (PK), EMP_NAME , HIRE_DATE
CUSTOMERS	CUST_ID (PK), CUST_NAME , ADDRESS
PRODUCTS	PROD_ID (PK), PROD_NAME , SUP_ID (FK), CAT_ID (FK), PRICE
ORDERS	ORD_ID (PK), CUST_ID (FK), EMP_ID (FK), ORDER_DATE
ORDER_DETAILS	ORD_DETID (PK, AUTO_INCREMENT), ORD_ID (FK), PROD_ID (FK), QUANTITY , EACH_PRICE , TOTAL_PRICE

PROBLEM STATEMENT

- ✓ **Efficient inventory, supplier, order, employee, and product category management are vital in the retail grocery industry.**
- ✓ **Poor data handling leads to issues in tracking sales, stock levels, and customer behavior.**
- ✓ **Operational inefficiencies arise with out structured data systems.**
- ✓ **This project simulates a mini grocery store database using SQL.**
- ✓ **Enables data-driven insights for informed decision-making.**
- ✓ **Supports performance optimization and business growth.**

DATA INSERTION AND MANIPULATION (DML)

INSERTION-

INSERTED 5 RECORDS FOR SUPPLIER TABLE AND 5 RECORDS FOR CATEGORIES TABLE

SUPPLIER TABLE

	sup_id	sup_name	address
▶	1	Aarav Sharma	33 Main Street, Madhya Pradesh, India
	2	Sai	108 Main Street, Telangana, India
	3	Aarya	166 Main Street, Uttar Pradesh, India
	4	Suresh	163 Main Street, Andhra Pradesh, India
	5	Karthik	182 Main Street, West Bengal, India
*	HULL	HULL	HULL

CATEGORIES

	cat_id	cat_name
▶	1	Grains & Cereals
	2	Dairy Products
	3	Beverages
	4	Personal Care
	5	Snacks & Confectioneries

DATA INSERTION AND MANIPULATION (DML)

INSERTION

**INSERTED 10 RECORDS FOR EMPLOYEE TABLE
AND 200 RECORDS CUSTOMERS TABLES**

EMPLOYEE

	emp_id	emp_name	hire_date
▶	1	Aarav Kumar 1	2/3/2021
	2	Aditya Singh 1	1/8/2021
	3	Pari Kumar 1	11/12/2021
	4	Aditya Verma 1	1/9/2021
	5	Pari Sharma 1	2/9/2021
	6	Zara Verma 1	10/16/2021
	7	Vihaan Singh 1	8/26/2020
	8	Diya Sharma 1	8/21/2021
	9	Arjun Kumar 1	5/29/2021
	10	Arjun Verma 1	4/14/2021

CUSTOMERS

	ord_id	cust_id	emp_id	order_date
▶	1	197	5	2022-01-30 00:00:00
	2	94	6	2022-07-02 00:00:00
	3	97	3	2022-11-25 00:00:00
	4	128	2	2022-05-04 00:00:00
	5	61	8	2022-03-05 00:00:00
	6	135	5	2022-08-17 00:00:00
	7	166	5	2022-04-22 00:00:00
	8	91	8	2022-03-03 00:00:00
	9	195	8	2022-10-31 00:00:00
	10	115	8	2022-11-30 00:00:00
	11	71	5	2022-06-18 00:00:00

DATA INSERTION AND MANIPULATION (DML)

INSERTION

**INSERTED 50 RECORDS FOR PRODUCTS TABLE AND
300 RECORDS FOR ORDERS TABLES**

PRODUCTS

	prod_id	prod_name	sup_id	cat_id	price
▶	1	Basmati Rice	3	1	358.98
	2	Wheat Flour	2	1	255.50
	3	Moong Dal	4	1	386.18
	4	Chickpeas	5	1	353.50
	5	Soybean Oil	3	1	172.81
	6	Ghee	3	1	487.46
	7	Paneer	2	2	484.27
	8	Yogurt	2	2	111.61
	9	Mango Pickle	5	1	182.50
	10	Mixed Vegetable Pickle	3	1	133.51
	11	Almonds	5	3	315.57

ORDERS

	ord_id	cust_id	emp_id	order_date
▶	1	197	5	2022-01-30 00:00:00
	2	94	6	2022-07-02 00:00:00
	3	97	3	2022-11-25 00:00:00
	4	128	2	2022-05-04 00:00:00
	5	61	8	2022-03-05 00:00:00
	6	135	5	2022-08-17 00:00:00
	7	166	5	2022-04-22 00:00:00
	8	91	8	2022-03-03 00:00:00
	9	195	8	2022-10-31 00:00:00
	10	115	8	2022-11-30 00:00:00
	11	71	5	2022-06-18 00:00:00

DATA INSERTION AND MANIPULATION (DML)

INSERTION

INSERTED 600 RECORDS FOR ORDER_DETAILS TABLE

ORDER_DETAILS

	ord_detID	ord_id	prod_id	quantity	each_price	total_price
▶	1	109	23	3	140.62	421.87
	2	144	12	1	441.95	441.95
	3	82	13	4	166.26	665.06
	4	224	18	2	219.36	438.73
	5	256	3	4	386.18	1544.71
	6	183	27	4	146.65	586.58
	7	174	26	3	464.02	1392.07
	8	164	42	1	322.40	322.40
	9	68	21	3	182.74	548.22
▶	10	129	3	1	386.18	386.18
	11	249	32	4	235.22	940.87

A photograph of a woman walking through a supermarket aisle, carrying a blue shopping basket. The shelves are stocked with various packaged snacks. The image is overlaid with a dark, semi-transparent circular graphic on the left side, which contains a white vertical bar. The text "CUSTOMER INSIGHTS" is written in large, bold, yellow capital letters across the center of the image.

CUSTOMER INSIGHTS

1A. HOW MANY UNIQUE CUSTOMERS HAVE PLACED ORDERS ?

```
select count(distinct cust_id)
from orders;
```

	count(distinct cust_id)
▶	156



1B. WHICH CUSTOMERS HAVE PLACE THE HIGHEST NUMBER OF ORDERS ?

```
select cust_id, count(*) as total_orders
from orders
group by cust_id
order by total_orders desc
limit 5;
```

	cust_id	total_orders
▶	165	7
	61	6
	19	5
	128	5
	145	5



1C. WHAT IS THE TOTAL AND AVERAGE PURCHASE VALUE PER CUSTOMER ?

```
select o.cust_id,  
       c.cust_name,  
       count(*) as total_orders_count,  
       sum(od.total_price) as total_orders_price,  
       avg(od.total_price) as average_orders_price  
from orders o join order_details od  
on o.ord_id = od.ord_id join customers c  
on o.cust_id = c.cust_id  
group by o.cust_id  
order by o.cust_id asc;
```

	cust_id	cust_name	total_orders_count	total_orders_price	average_orders_price
▶	1	Aditi Shetty	1	1577.86	1577.860000
	2	Isha Reddy	2	1299.62	649.810000
	3	Chetan Rao	9	7693.41	854.823333
	5	Isha Rao	2	3327.05	1663.525000
	7	Eshwar Iyer	14	9188.45	656.317857
	8	Deepa Reddy	9	7929.13	881.014444
	13	Bala Krishnan	2	1161.66	580.830000
	14	Jaya Shetty	2	1932.94	966.470000
	15	Aditi Gowda	2	948.15	474.075000



1D. WHO ARE THE TOP 5 CUSTOMERS BY TOTAL PURCHASE AMOUNT ?

```
select o.cust_id,  
       c.cust_name,  
       sum(od.total_price) as total_orders_price  
from orders o join order_details od  
on o.ord_id = od.ord_id join customers c  
on o.cust_id = c.cust_id  
group by o.cust_id  
order by total_orders_price desc  
limit 5;
```

	cust_id	cust_name	total_orders_price
▶	19	Chetan Naidu	11256.82
	166	Kapila	11099.51
	67	Eshwar Rao	10819.96
	61	Aditi Rao	10230.64
	7	Eshwar Iyer	9188.45





PRODUCT PERFORMANCE

2A. HOW MANY PRODUCTS EXIST IN EACH CATEGORY

```
SELECT c.cat_name, COUNT(*) AS product_count
FROM products p
JOIN categories c ON p.cat_id = c.cat_id
GROUP BY c.cat_name
ORDER BY product_count DESC;
```



	cat_name	product_count
▶	Grains & Cereals	18
	Beverages	17
	Dairy Products	6
	Personal Care	6
	Snacks & Confectioneries	3

2B. AVERAGE PRODUCT PRICE BY CATEGORY

```
SELECT c.cat_name,  
       AVG(p.price) AS avg_price_per_category  
FROM products p JOIN categories c  
ON p.cat_id = c.cat_id  
GROUP BY c.cat_name  
ORDER BY avg_price_per_category DESC;
```

	cat_name	avg_price_per_category
►	Dairy Products	366.943333
	Personal Care	364.991667
	Snacks & Confectioneries	363.336667
	Grains & Cereals	287.673333
	Beverages	278.892353



2C. PRODUCTS WITH HIGHEST SALES VOLUME



```
SELECT p.prod_name, SUM(od.quantity) AS total_quantity
FROM order_details od
JOIN products p ON od.prod_id = p.prod_id
GROUP BY p.prod_name
ORDER BY total_quantity DESC
LIMIT 5;
```

	prod_name	total_quantity
▶	Bath Soap	60
	Hand Sanitizer	56
	Dishwashing Soap	54
	Biscuits	54
	Potato Chips	54



2D. REVENUE GENERATED BY EACH PRODUCTS

```
SELECT p.prod_name, SUM(od.total_price) AS total_revenue
FROM order_details od
JOIN products p ON od.prod_id = p.prod_id
GROUP BY p.prod_name
ORDER BY total_revenue DESC;
```

Result Grid |   Filter Rows:

	prod_name	total_revenue
▶	Hand Sanitizer	27787.76
	Biscuits	20995.92
	Moong Dal	19695.02
	Toothpaste	19688.95
	Mustard Seeds	19516.68
	Cashews	18561.92
	Butter	18548.40
	Cheese Slices	18519.61
	Turmeric Powder	17784.29
	Soya Sauce	16985.38



2E. PRODUCT SALES BY CATEGORY AND SUPPLIER

```
SELECT c.cat_name, s.sup_name,  
       SUM(od.total_price) AS total_revenue  
FROM order_details od  
JOIN products p ON od.prod_id = p.prod_id  
JOIN categories c ON p.cat_id = c.cat_id  
JOIN supplier s ON p.sup_id = s.sup_id  
GROUP BY c.cat_name, s.sup_name  
ORDER BY total_revenue DESC;
```



	cat_name	sup_name	total_revenue
▶	Personal Care	Aarya	69378.41
	Grains & Cereals	Aarya	67701.10
	Beverages	Aarya	65538.71
	Beverages	Suresh	65307.14
	Dairy Products	Sai	50740.60
	Grains & Cereals	Karthik	39473.49
	Beverages	Aarav Sharma	26948.15
	Grains & Cereals	Suresh	26248.89
	Snacks & Confectioneries	Karthik	22767.59

The background is a dark blue gradient with a grid of dashed white lines. It features several abstract financial charts: a line graph with a peak labeled '12.0098' and a trough labeled '0.0103', a bar chart with teal bars, and a large, 3D-style upward-pointing arrow in the foreground. The text 'SALES & ORDER TRENDS' is centered in a bold, yellow, sans-serif font.

SALES & ORDER TRENDS

CHALLENGE WE FACED IN PROJECT

Extracting time-based trends from date data (especially if in VARCHAR format).

```
UPDATE orders SET order_date = STR_TO_DATE(order_date, '%m/%d/%Y');  
ALTER TABLE orders MODIFY COLUMN order_date DATETIME;
```

3A. HOW MANY ORDERS HAVE BEEN PLACED IN TOTAL ?

```
select count(*)  
from order_details;
```

	count(*)
▶	600

3B. WHAT IS THE AVERAGE VALUE PER ORDER ?

```
select od.ord_id,  
       avg(od.total_price) as average_orders_price  
from orders o join order_details od  
on o.ord_id = od.ord_id  
group by od.ord_id  
order by average_orders_price desc  
limit 5;
```



	ord_id	average_orders_price
▶	155	2497.850000
	20	2437.280000
	14	2421.340000
	69	2237.380000
	88	2237.380000

3C. ON WHICH DATES WERE THE MOST ORDERS PLACED ?

```
select date(order_date),  
       count(*) as orders_count  
from orders  
group by order_date  
order by orders_count desc  
limit 8;
```

	date(order_date)	orders_count
►	2022-09-10	4
	2022-03-30	4
	2022-01-30	3
	2022-01-16	3
	2022-01-14	3
	2022-06-27	3
	2022-04-22	3
	2022-12-05	3



3D. WHAT ARE THE MONTHLY TRENDS IN ORDER VOLUME AND REVENUE ?

```
select monthname(o.order_date) as order_month,  
       count(*) as total_orders_count,  
       sum(od.quantity) as volume,  
       sum(od.total_price) as revenue  
from order_details od join orders o  
on od.ord_id = o.ord_id  
group by order_month  
order by revenue desc;
```

	order_month	total_orders_count	volume	revenue
▶	January	79	227	70312.45
	February	66	214	66929.42
	December	62	200	60903.12
	September	57	175	52626.61
	July	50	148	48674.66
	November	47	137	46141.33
	March	57	163	45977.16
	May	46	126	41305.62
	August	41	112	36045.01
	April	32	101	29118.54
	June	31	81	27378.69
	October	32	88	25917.32



3E. HOW DO ORDER PATTERNS VARY ACROSS WEEK DAYS AND MONTHS ?



```
select monthname(o.order_date) as order_month,  
       dayname(o.order_date) as order_weekday,  
       count(*) as total_orders_count,  
       sum(od.quantity) as total_orders_quantity,  
       sum(od.total_price) as total_orders_price  
from order_details od join orders o  
on od.ord_id = o.ord_id  
group by monthname(o.order_date), dayname(o.order_date)  
limit 10;
```

	order_month	order_weekday	total_orders_count	total_orders_quantity	total_orders_price
▶	July	Saturday	16	48	15806.56
	November	Friday	15	40	12870.05
	May	Wednesday	3	5	1079.65
	March	Saturday	7	23	6513.11
	August	Wednesday	14	44	14758.99
	April	Friday	12	42	12454.74
	March	Thursday	18	43	12187.95
	October	Monday	6	19	6903.69
	November	Wednesday	8	28	7042.26
	August	Tuesday	1	5	1577.86



SUPPLIER CONTRIBUTION

4A. HOW MANY SUPPLIERS ARE THEIR IN THE DATA BASE ?

```
select * from supplier;
```

	sup_id	sup_name	address
▶	1	Aarav Sharma	33 Main Street, Madhya Pradesh, India
	2	Sai	108 Main Street, Telangana, India
	3	Aarya	166 Main Street, Uttar Pradesh, India
	4	Suresh	163 Main Street, Andhra Pradesh, India
	5	Karthik	182 Main Street, West Bengal, India
★	NULL	NULL	NULL



4B. WHICH SUPPLIER PROVIDES THE MOST PRODUCTS?



```
select p.sup_id,  
       s.sup_name,  
       count(*) as total_products_count  
from products p join supplier s  
on p.sup_id = s.sup_id  
group by s.sup_id  
order by total_products_count desc;
```

	sup_id	sup_name	total_products_count
▶	3	Aarya	18
	2	Sai	10
	4	Suresh	10
	5	Karthik	9
	1	Aarav Sharma	3

4C. WHAT IS THE AVERAGE PRICE OF PRODUCTS FROM EACH SUPPLIER?


```
select p.sup_id,  
       s.sup_name,  
       avg(p.price) as average_price_supplier  
from products p join supplier s  
on p.sup_id = s.sup_id  
group by s.sup_id  
order by average_price_supplier desc;
```

	sup_id	sup_name	average_price_supplier
▶	2	Sai	342.672000
	3	Aarya	319.326667
	5	Karthik	288.225556
	4	Suresh	281.818000
	1	Aarav Sharma	271.366667



4D. WHICH SUPPLIERS CONTRIBUTE THE MOST TO TOTAL PRODUCT SALES (BY REVENUE)?

```
select p.sup_id,  
       s.sup_name,  
       sum(price) as revenue  
from products p join supplier s  
on p.sup_id = s.sup_id  
group by s.sup_id  
order by revenue desc;
```



	sup_id	sup_name	revenue
▶	3	Aarya	5747.88
	2	Sai	3426.72
	4	Suresh	2818.18
	5	Karthik	2594.03
	1	Aarav Sharma	814.10



EMPLOYEE PERFORMANCE

5A. HOW MANY EMPLOYEES HAVE PROCESSED ORDERS?

```
SELECT COUNT(DISTINCT emp_id) AS employees_processed_orders  
FROM Orders;
```

	employees_processed_orders
▶	10



5B. WHICH EMPLOYEES HAVE HANDLED THE MOST ORDERS?

```
select o.emp_id,  
       e.emp_name,  
       count(*) as orders_count  
from orders o join employees e  
on o.emp_id = e.emp_id  
group by o.emp_id  
order by orders_count desc;
```

	emp_id	emp_name	orders_count
►	8	Diya Sharma 1	38
	2	Aditya Singh 1	37
	9	Arjun Kumar 1	32
	3	Pari Kumar 1	31
	5	Pari Sharma 1	31
	6	Zara Verma 1	30
	7	Vihaan Singh 1	29
	4	Aditya Verma 1	26
	1	Aarav Kumar 1	23
	10	Arjun Verma 1	23



5C. WHAT IS THE TOTAL SALES VALUE PROCESSED BY EACH EMPLOYEE?

```
select o.emp_id,  
       e.emp_name,  
       sum(od.total_price) as total_sales  
from order_details od join orders o  
on od.ord_id = o.ord_id join employees e  
on o.emp_id = e.emp_id  
group by o.emp_id  
order by total_sales desc;
```

	emp_id	emp_name	total_sales
▶	2	Aditya Singh 1	79252.29
	6	Zara Verma 1	71562.76
	8	Diya Sharma 1	67241.85
	3	Pari Kumar 1	66818.39
	9	Arjun Kumar 1	54018.31
	1	Aarav Kumar 1	52602.88
	7	Vihaan Singh 1	48577.88
	5	Pari Sharma 1	40334.22
	10	Arjun Verma 1	36716.84
	4	Aditya Verma 1	34204.51



5D. WHAT IS THE AVERAGE ORDER VALUE HANDLED PER EMPLOYEE?

```
select e.emp_id,  
       e.emp_name,  
       avg(total_price) as average_price  
from order_details od left join orders o  
on od.ord_id = o.ord_id join employees e  
on o.emp_id = e.emp_id  
group by o.emp_id;
```

	emp_id	emp_name	average_price
▶	1	Aarav Kumar 1	1073.528163
	2	Aditya Singh 1	1003.193544
	3	Pari Kumar 1	856.646026
	4	Aditya Verma 1	760.100222
	5	Pari Sharma 1	840.296250
	6	Zara Verma 1	941.615263
	7	Vihaan Singh 1	916.563774
	8	Diya Sharma 1	988.850735
	9	Arjun Kumar 1	915.564576
	10	Arjun Verma 1	815.929778



ORDER DETAILS DEEP DIVE

6A. WHAT IS THE RELATIONSHIP BETWEEN QUANTITY ORDERED AND TOTAL PRICE ?

```
select quantity, sum(total_price) as total_quantity_price  
from order_details  
group by quantity  
order by total_quantity_price asc, quantity asc;
```

	quantity	total_quantity_price
▶	1	39590.04
	2	69698.85
	3	123960.76
	4	140553.75
	5	177526.53



6B. WHAT IS THE AVERAGE QUANTITY ORDERED PER PRODUCT ?

```
select od.prod_id,  
       p.prod_name,  
       avg(od.quantity)  
from order_details od join products p  
on od.prod_id = p.prod_id  
group by prod_id;
```

	prod_id	prod_name	avg(od.quantity)
▶	1	Basmati Rice	3.2000
	2	Wheat Flour	2.5333
	3	Moong Dal	3.4000
	4	Chickpeas	2.4286
	5	Soybean Oil	1.6364
	6	Ghee	3.3750
	7	Paneer	3.0000



6C. HOW DOES THE UNIT PRICE VARY ACROSS PRODUCTS AND ORDERS ?

```
select ord_id,  
       prod_id,  
       each_price  
from order_details  
order by prod_id asc, ord_id asc;
```

	ord_id	prod_id	each_price
▶	31	1	358.98
	36	1	358.98
	56	1	358.98
	70	1	358.98
	93	1	358.98
	112	1	358.98
	143	1	358.98
	240	1	358.98
	266	1	358.98
	292	1	358.98
	6	2	255.50
	21	2	255.50

CONCLUSION

- **Created a database system** with key tables like Customers, Products, Orders, and Suppliers to organize business data.
- **Used primary and foreign keys** to keep the data accurate and properly linked between tables.
- **Combined data from different tables** using SQL joins to answer complex questions.
- **Used functions to calculate totals and summaries**, like total sales and product stock levels.
- **Helped the business make better decisions** by showing clear information on customer purchases and popular products.

**HAPPY
CUSTOMER**



THANK YOU

