Prepared by Reshma

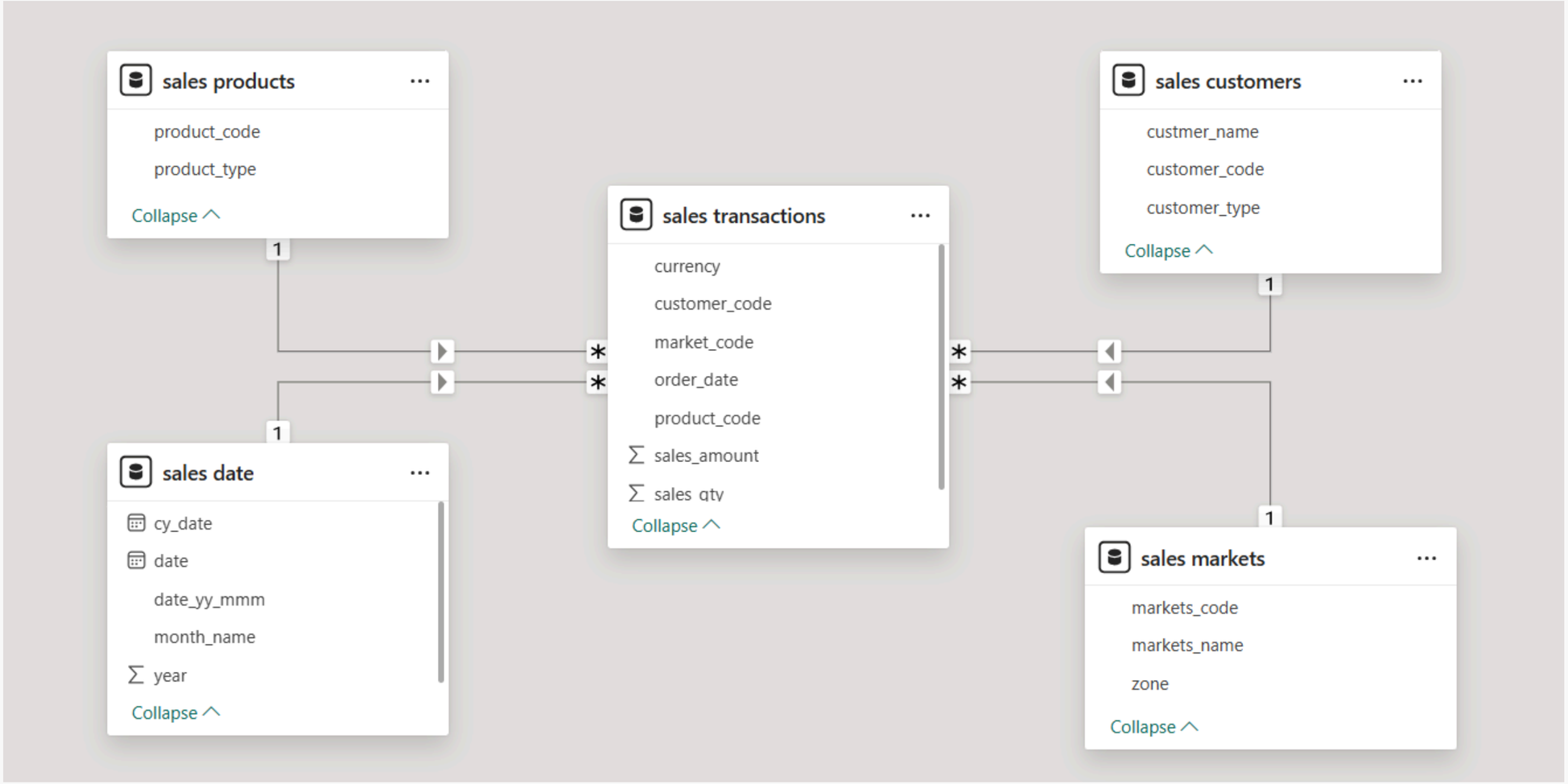
Sales Insights Project

17 April, 2025

Introduction

This project involves analyzing the sales performance of Atliq Hardware, a B2B consumer electronics manufacturer, using SQL. The goal is to extract meaningful business insights by querying a relational database containing sales, customer, product, and market data. Using MySQL, I explored trends in revenue, market performance, and customer engagement to support strategic decision-making and uncover opportunities for growth and optimization.

Data Model


sales products
product_code
product_type
Collapse
sales transactions
currency
customer_code
market_code
order_date
product_code
sales_amount
sales_qty
Collapse
sales customers
custmer_name
customer_code
customer_type
Collapse
sales date
cy_date
date
date_yy_mmm
month_name
year
Collapse
sales markets
markets_code
markets_name
zone
Collapse
1
1
1
1
*
*
*
*

Data Model (continuation)

This star schema model centers around the sales_transactions fact table, which records transactional data such as sales amount, quantity, currency, and references to dimensions (date, customer, product, and market).

It is connected to four dimension tables that provide descriptive attributes:

- sales_date: Contains date-related information such as calendar year (year), month name, and custom formatted date fields. Linked via order_date.
- sales_customers: Holds customer details like customer_name, customer_type, and customer_code.
- sales_products: Includes product_code and product_type, offering product-level granularity.
- sales_markets: Provides market-level data including markets_name and zone.

Each dimension table is connected to the central sales_transactions table using one-to-many relationships, supporting efficient querying and drill-downs in visual dashboards. This model enables robust analysis of performance trends, customer behavior, market dynamics, and time-based sales metrics.

All transactions made from Chennai

```sql
-- All transactions made from Chennai
SELECT
    *
FROM
    transactions
WHERE
    market_code = 'Mark001';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency
Prod001	Cus001	Mark001	2017-10-10	100	41241	INR
Prod013	Cus001	Mark001	2017-10-10	240	143560	INR
Prod001	Cus001	Mark001	2017-10-10	100	41241	INR
Prod013	Cus001	Mark001	2017-10-10	240	143560	INR
Prod016	Cus001	Mark001	2018-06-28	187	115481	INR
Prod020	Cus001	Mark001	2017-10-10	140	153019	INR
Prod020	Cus001	Mark001	2017-10-25	47	51005	INR
Prod020	Cus001	Mark001	2017-11-14	47	51005	INR
Prod040	Cus001	Mark001	2018-01-05	240	168245	INR
Prod040	Cus001	Mark001	2018-01-12	167	104648	INR
Prod040	Cus001	Mark001	2018-01-15	240	168245	INR
Prod040	Cus001	Mark001	2018-01-30	193	135532	INR
Prod040	Cus001	Mark001	2018-02-12	40	28042	INR
Prod040	Cus001	Mark001	2018-02-13	40	25116	INR
Prod040	Cus001	Mark001	2018-04-03	480	336495	INR
Prod040	Cus001	Mark001	2018-04-27	120	75347	INR
Prod040	Cus001	Mark001	2018-04-30	120	84125	INR
Prod040	Cus001	Mark001	2018-05-16	160	100463	INR
Prod040	Cus001	Mark001	2018-08-15	400	251162	INR
Prod040	Cus001	Mark001	2018-10-11	40	28042	INR
Prod040	Cus001	Mark001	2019-05-28	80	50231	INR

All transactions made in USD currency

```
27      -- All transactions made in USD currency
28  •   SELECT
29          *
30      FROM
31          transactions
32      WHERE
33          currency = 'USD';
34
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency
Prod003	Cus005	Mark004	2017-11-20	59	500	USD
Prod003	Cus005	Mark004	2017-11-22	36	250	USD

Total revenue or sales made in the year 2020

```sql
-- Total revenue or sales made in the year 2020
SELECT
    SUM(sales_amount) AS total_revenue
FROM
    transactions
        JOIN
    `date` ON transactions.order_date = `date`.`date`
WHERE
    `date`.year = '2020';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_revenue
142235559

Total business done in chennai the year 2020

```
55      -- Total business done in chennai in the year 2020
56  •   SELECT
57          SUM(sales_amount) AS total_revenue
58      FROM
59          transactions
60              JOIN
61          `date` ON transactions.order_date = `date`.`date`
62      WHERE
63          `date`.year = '2020' and transactions.market_code = 'Mark001';
64
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	total_revenue
▶	2463024

Distinct products sold in Chennai

```
64
65      -- Distinct products sold in chennai
66 •    SELECT DISTINCT
67          (product_code)
68      FROM
69          transactions
70      WHERE
71          transactions.market_code = 'Mark001';
72
73
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_code
Prod001
Prod013
Prod016
Prod020
Prod040
Prod044
Prod061
Prod062
Prod065
Prod070
Prod078
Prod090
Prod101
Prod105
Prod106
Prod107
Prod110
Prod113

List all unique customer types

```sql
-- List all unique customer types.
SELECT DISTINCT
    (customer_code)
FROM
    customers;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_code
Cus001
Cus002
Cus003
Cus004
Cus005
Cus006
Cus007
Cus008
Cus009
Cus010
Cus011
Cus012
Cus013
Cus014
Cus015
Cus016
Cus017
Cus018
Cus019
Cus020

Get the total number of customers in each customer type

```sql
-- Get the total number of customers in each customer type.
SELECT
    customer_type, COUNT(*) AS total_customers
FROM
    customers
GROUP BY customer_type;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_type	total_customers
Brick & Mortar	19
E-Commerce	19

Display all transactions that happened in the year 2020

```sql
-- Display all transactions that happened in the year 2020.
SELECT
    *
FROM
    transactions
        JOIN
    `date` ON transactions.order_date = `date`.`date`
WHERE
    `date`.year = '2020';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	date	cy_date	year	month_name	date_yy_mmm
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020-01-10	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020-01-17	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020-02-07	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020-02-14	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020-02-28	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-03-06	1	620	INR	2020-03-06	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-03-13	1	620	INR	2020-03-13	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-04-03	1	829	INR	2020-04-03	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-14	4	2694	INR	2020-04-14	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-20	1	102	INR	2020-04-20	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-05-15	2	528	INR	2020-05-15	2020-05-01	2020	May	20-May
Prod011	Cus016	Mark002	2020-06-12	1	1028	INR	2020-06-12	2020-06-01	2020	June	20-Jun
Prod011	Cus016	Mark002	2020-06-16	1	514	INR	2020-06-16	2020-06-01	2020	June	20-Jun
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan

Retrieve the names of all markets in the North zone

```sql
-- Retrieve the names of all markets in the 'North' zone.
SELECT
    markets_name
FROM
    markets
WHERE
    zone = 'North';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	date	cy_date	year	month_name	date_yy_mmm
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020-01-10	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020-01-17	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020-02-07	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020-02-14	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020-02-28	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-03-06	1	620	INR	2020-03-06	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-03-13	1	620	INR	2020-03-13	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-04-03	1	829	INR	2020-04-03	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-14	4	2694	INR	2020-04-14	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-20	1	102	INR	2020-04-20	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-05-15	2	528	INR	2020-05-15	2020-05-01	2020	May	20-May
Prod011	Cus016	Mark002	2020-06-12	1	1028	INR	2020-06-12	2020-06-01	2020	June	20-Jun
Prod011	Cus016	Mark002	2020-06-16	1	514	INR	2020-06-16	2020-06-01	2020	June	20-Jun
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020-01-10	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020-01-17	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020-02-07	2020-02-01	2020	February	20-Feb

Show all products and their product types

```
32      -- Show all products and their product types.
33  •   SELECT
34          *
35      FROM
36          products;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	date	cy_date	year	month_name	date_yy_mmm
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020-01-10	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020-01-17	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020-02-07	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020-02-14	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020-02-28	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-03-06	1	620	INR	2020-03-06	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-03-13	1	620	INR	2020-03-13	2020-03-01	2020	March	20-Mar
Prod005	Cus007	Mark004	2020-04-03	1	829	INR	2020-04-03	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-14	4	2694	INR	2020-04-14	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-04-20	1	102	INR	2020-04-20	2020-04-01	2020	April	20-Apr
Prod005	Cus007	Mark004	2020-05-15	2	528	INR	2020-05-15	2020-05-01	2020	May	20-May
Prod011	Cus016	Mark002	2020-06-12	1	1028	INR	2020-06-12	2020-06-01	2020	June	20-Jun
Prod011	Cus016	Mark002	2020-06-16	1	514	INR	2020-06-16	2020-06-01	2020	June	20-Jun
Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020-01-09	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020-01-10	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020-01-17	2020-01-01	2020	January	20-Jan
Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020-02-07	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020-02-14	2020-02-01	2020	February	20-Feb
Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020-02-28	2020-02-01	2020	February	20-Feb

What is the total sales amount and quantity by customer for the month of January 2020

```sql
-- What is the total sales amount and quantity by customer for the month of January 2024?
SELECT
    customers.custmer_name,
    SUM(transactions.sales_qty) AS total_qty,
    SUM(transactions.sales_amount) AS total_amount
FROM
    transactions
        JOIN
    `date` ON transactions.order_date = `date`.`date`
        JOIN
    customers ON transactions.customer_code = customers.customer_code
WHERE
    `date`.year = 2020
        AND `date`.month_name = 'January'
GROUP BY customers.custmer_name
order by total_amount desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

custmer_name	total_qty	total_amount
Electricalsara Stores	19298	13386461
Electricalslytical	2199	1269589
Excel Stores	2459	935980
Premium Stores	4372	895498
Forward Stores	1153	675038
Control	1322	640294
Acclaimed Stores	743	632420

Find the top 5 customers by total sales amount

```sql
-- Find the top 5 customers by total sales amount.
SELECT
    customers.custmer_name, SUM(sales_amount) AS total_sales
FROM
    transactions
        JOIN
    customers ON transactions.customer_code = customers.customer_code
GROUP BY customers.custmer_name
ORDER BY total_sales desc
limit 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

custmer_name	total_qty	total_amount
Electricalsara Stores	19298	13386461
Electricalslytical	2199	1269589
Excel Stores	2459	935980
Premium Stores	4372	895498
Forward Stores	1153	675038
Control	1322	640294
Acclaimed Stores	743	632420
Nixon	2907	620018
Info Stores	1464	609769
Surge Stores	4290	601875
Modular	1975	525586
Atlas Stores	771	453881
Epic Stores	1621	448911
Surface Stores	2739	443331

Which market had the highest sales in 2020

```sql
-- Which market had the highest sales in 2023?
SELECT
    markets.markets_name, SUM(sales_amount) AS total_sales
FROM
    transactions
        JOIN
    markets ON transactions.market_code = markets.markets_code
        JOIN
    `date` ON transactions.order_date = `date`.`date`
WHERE
    `date`.year = '2020'
    group by markets.markets_name
    order by total_sales desc
    limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

markets_name	total_sales
Delhi NCR	77742074

Get the total sales amount per product type

```
80
81      -- Get the total sales amount per product type.
82      SELECT
83          products.product_type, SUM(sales_amount) AS total_sales
84      FROM
85          transactions
86              JOIN
87          products ON transactions.product_code = products.product_code
88          group by products.product_type
89          order by total_sales desc;
90
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_type	total_sales
Own Brand	371564951
Distribution	146039576

List all customers who made purchases in more than 3 different markets

```sql
-- List all customers who made purchases in more than 3 different markets.
SELECT
    customers.custmer_name, count(distinct transactions.market_code) AS markets_visited
FROM
    transactions
        JOIN
    customers ON transactions.customer_code = customers.customer_code
GROUP BY customers.custmer_name
having count(distinct transactions.market_code) >3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

custmer_name	markets_visited
Control	4
Electricalsara Stores	4
Electricalsbea Stores	4
Electricalslance Stores	4
Epic Stores	4
Excel Stores	4
Flawless Stores	4
Info Stores	4
Nixon	4
Nomad Stores	4
Sound	4
Surface Stores	4
Unity Stores	4

Join all tables to get a full view

```sql
-- Join all tables to get a full view: date, product, customer, market, sales quantity, and sales amount for each transaction.
SELECT
    transactions.*,
    customers.custmer_name,
    customers.customer_type,
    `date`.cy_date,
    `date`.year,
    `date`.month_name,
    `date`.date_yy_mmm,
    markets.markets_name,
    markets.zone,
    products.product_type
FROM
    transactions
        JOIN
    customers ON transactions.customer_code = customers.customer_code
        JOIN
    `date` ON transactions.order_date = `date`.`date`
        JOIN
    markets ON transactions.market_code = markets.markets_code
```

Join all tables to get a full view

```
121          JOIN
122      products ON transactions.product_code = products.product_code;
123
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	custmer_name	customer_type	cy_date	year	month_name	date_yy_mmm	mar
Prod001	Cus001	Mark001	2017-10-10	100	41241	INR	Surge Stores	Brick & Mortar	2017-10-01	2017	October	17-Oct	Che
Prod001	Cus002	Mark002	2018-05-08	3	-1	INR	Nomad Stores	Brick & Mortar	2018-05-01	2018	May	18-May	Mum
Prod002	Cus003	Mark003	2018-04-06	1	875	INR	Excel Stores	Brick & Mortar	2018-04-01	2018	April	18-Apr	Ahm
Prod002	Cus003	Mark003	2018-04-11	1	583	INR	Excel Stores	Brick & Mortar	2018-04-01	2018	April	18-Apr	Ahm
Prod002	Cus004	Mark003	2018-06-18	6	7176	INR	Surface Stores	Brick & Mortar	2018-06-01	2018	June	18-Jun	Ahm
Prod003	Cus005	Mark004	2017-11-20	59	500	USD	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-22	36	250	USD	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-23	39	21412	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-27	35	19213	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-28	310	170185	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-29	184	101194	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod003	Cus005	Mark004	2017-11-30	35	19213	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod004	Cus005	Mark004	2017-11-29	17	9426	INR	Premium Stores	Brick & Mortar	2017-11-01	2017	November	17-Nov	Delh
Prod004	Cus005	Mark004	2017-12-19	1	218	INR	Premium Stores	Brick & Mortar	2017-12-01	2017	December	17-Dec	Delh
Prod005	Cus005	Mark004	2018-08-07	5	3093	INR	Premium Stores	Brick & Mortar	2018-08-01	2018	August	18-Aug	Delh

Show total sales by zone and year

```sql
-- Show total sales by zone and year.
SELECT
    markets.zone, `date`.year, SUM(sales_amount) AS total_sales
FROM
    transactions
        JOIN
    markets ON transactions.market_code = markets_code
        JOIN
    `date` ON transactions.order_date = `date`.`date`
GROUP BY markets.zone , `date`.year
ORDER BY total_sales DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

zone	year	total_sales
North	2018	287575003
North	2019	225622552
Central	2018	107923821
North	2020	99575269
Central	2019	95374857
North	2017	64187166
Central	2020	36099770
Central	2017	24462564
South	2018	18810117
South	2019	15454705
South	2020	6560520
South	2017	4919422

Compare monthly sales trends by product type across the last two years

```sql
-- Compare monthly sales trends by product type across the last two years.
SELECT
    `date`.year,
    `date`.month_name,
    products.product_type,
    SUM(transactions.sales_amount) as total_sales
FROM
    transactions
        JOIN
    `date` ON transactions.order_date = `date`.`date`
        JOIN
    products ON products.product_code = transactions.product_code
WHERE
    `date`.year IN (2019 , 2020)
GROUP BY `date`.year , `date`.month_name , products.product_type
ORDER BY `date`.year , `date`.month_name , products.product_type;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

year	month_name	product_type	total_sales
2019	April	Distribution	5055344
2019	April	Own Brand	9932020
2019	August	Distribution	5234753
2019	August	Own Brand	12675435
2019	December	Distribution	4649595
2019	December	Own Brand	8292830
2019	February	Distribution	3455548
2019	February	Own Brand	9971049
2019	January	Distribution	3540126
2019	January	Own Brand	10434219

Which product type had the highest average sales amount per transaction

```sql
-- Which product type had the highest average sales amount per transaction?
SELECT
    products.product_type,
    ROUND(AVG(sales_amount), 2) AS Average_sales
FROM
    transactions
        JOIN
    products ON transactions.product_code = products.product_code
GROUP BY products.product_type
ORDER BY Average_sales DESC
LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_type	Average_sales
Own Brand	6067.46

For each market list the top selling customer (by sales amount)

```sql
-- For each market, list the top-selling customer (by sales amount).
select markets_name, custmer_name, total_sales from
(SELECT
    markets.markets_name,
    customers.custmer_name,
    SUM(transactions.sales_amount) AS total_sales,
    rank() over(partition by transactions.market_code order by SUM(transactions.sales_amount) desc) as rnk
FROM
    transactions

        JOIN
    markets ON markets.markets_code = transactions.market_code
        JOIN
    customers ON customers.customer_code = transactions.customer_code
GROUP BY   markets.markets_name , customers.custmer_name, transactions.market_code
)sub
where rnk = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

markets_name	custmer_name	total_sales
Chennai	Surge Stores	18227503
Mumbai	Electricalslytical	48095673
Ahmedabad	Excel Stores	31481318
Delhi NCR	Electricalsara Stores	411108413
Kanpur	Acclaimed Stores	7861938
Bengaluru	Flawless Stores	290750
Bhopal	Epic Stores	10237161
Lucknow	Insight	3094007
Patna	Elite	4428393

Rank customers by sales amount within each zone

```sql
-- Rank customers by sales amount within each zone.
SELECT
    customers.custmer_name,
    markets.zone,
    SUM(transactions.sales_amount) AS total_sales,
    rank() over (partition by markets.zone order by  SUM(transactions.sales_amount) desc) as rnk
FROM
    transactions
    join customers on customers.customer_code = transactions.customer_code
    join markets on markets_code = transactions.market_code
    group by  customers.custmer_name, markets.zone;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

custmer_name	zone	total_sales	rnk
Electricalslytical	Central	49644189	1
Epic Stores	Central	18745111	2
Electricalsocity	Central	17489935	3
Modular	Central	16909638	4
Nomad Stores	Central	16835050	5
Leader	Central	16529970	6
Acclaimed Stores	Central	13336103	7
Logic Stores	Central	13264523	8
Path	Central	12995938	9
Electricalsopedia Stores	Central	10310851	10
Excel Stores	Central	10250431	11
Nixon	Central	7216897	12

Thank you