

IE 6700

Case Study

Group 22

Executive Summary

Reliance Industries recently announced that it has entered into a master franchise agreement with the Texas-based company 7-Eleven. The deal is aimed at launching 7-Eleven convenience stores in India. To fully utilize the massive Indian market to make a great start, Reliance will share some of its Supply-Chain and Customer-centric data with 7-Eleven so that it can quickly adapt to the Indian market. The Supply-Chain data will ensure a smooth demand and supply whereas the Customer-centric data will facilitate 7-Eleven to provide targeted ads for Indian consumers. The goal is to build a database that can accommodate the above requirements and ensure all the data is stored systematically and queries can be made to extract information smoothly.

Introduction

Superstores are a huge part of the foodservice industry. A superstore is a grocery store with more than the average amount of space and variety of stock. Thus a customer gets to choose from a variety of products while shopping for their daily groceries. The two superstores taken into consideration are reliance fresh and 7-eleven. Reliance fresh is a well-established superstore chain in India while similarly 7-eleven is well established in the United States. In the first week of October 2021, these two brands announced their merger which meant that Reliance planned on opening 7-eleven stores in India. The logistics, operations management, and cash-flow mechanism of a convenience store business is markedly different from regular modern retail stores and varies from country to country as well. Thus even though 7-eleven has a huge variety of products to offer to the Indian market, the logistics, operations management, and cash flow mechanism differs significantly from that in the United States. This essential data will be provided by reliance on fresh. Thus the relational database of reliance fresh will be accessed by 7-eleven to understand these aspects of the Indian market. But reliance fresh won't be willing to share its entire database with them. Thus the administrator selectively shares the database such that they understand the mechanisms of the market and at the same time no sensitive information is passed on to 7-eleven as well. The database will also give them insights into the scope of their product in the Indian market and where the product should be first introduced in the country. This is very crucial information as there is a possibility that some products of particular categories might not be as successful as their competitors that already exist in the market. The database will also give them information about the trends in the market. This database allows 7-eleven to establish its roots in the Indian market more efficiently.

In implementing this database, the following rules must be followed:

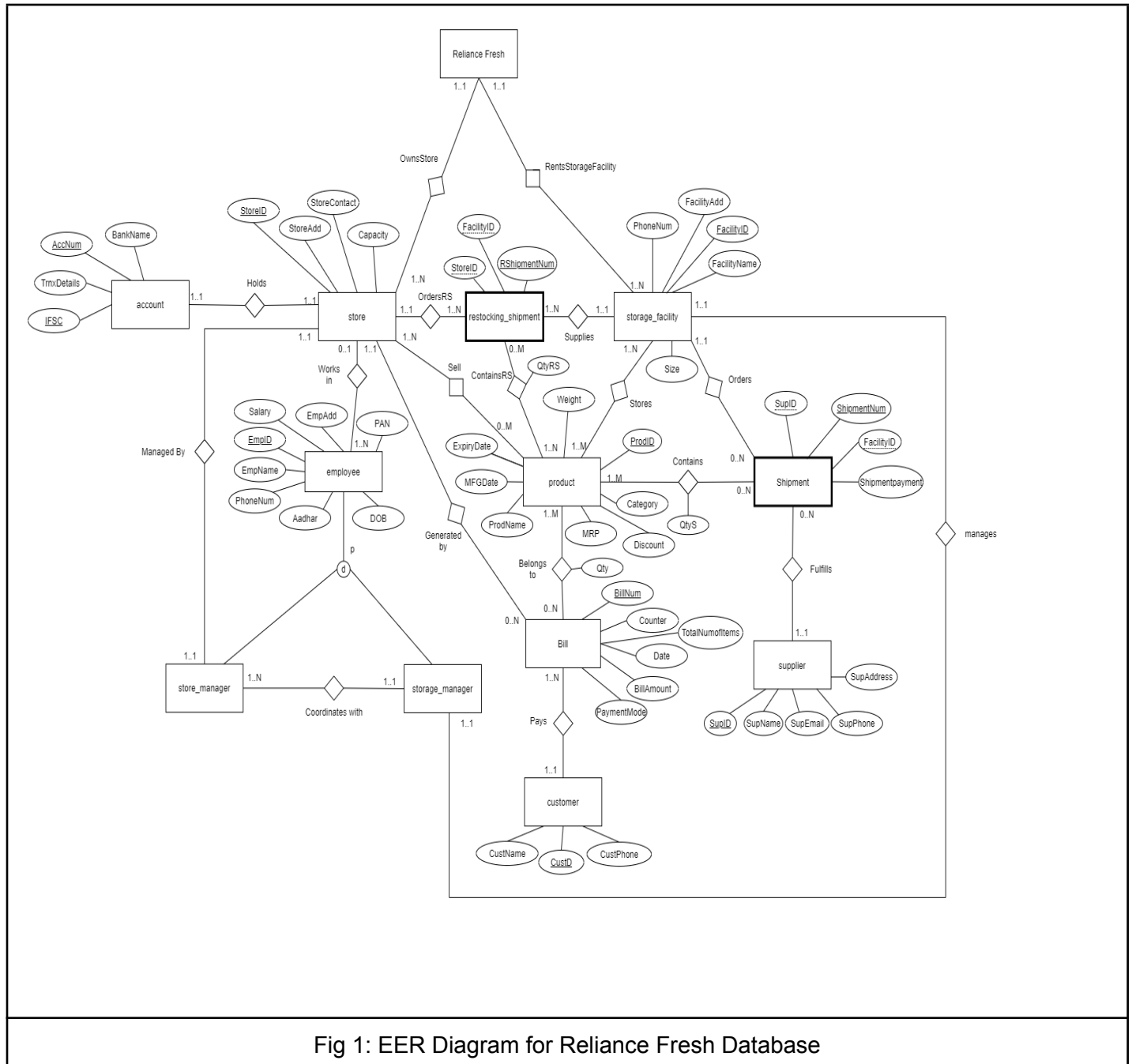
- Reliance Fresh owns several stores and storage facilities.
- Each store has many employees but one and only one store manager.

- A storage manager is assigned to a storage facility that has been rented by Reliance Fresh. The store manager coordinates with one and only one storage manager, while a storage manager has to coordinate with more than one store manager.
- Each customer can pay multiple bills at a store that can have multiple products with varying quantities.
- Each store has an account associated with a bank.
- Each store has a variety of products. Each time the products need to be restocked, the storage facility supplies a restocking shipment.
- The supplier supplies a shipment to the storage facility. The shipment has the required quantity of one or more products.
- A shipment is fulfilled by one and only one supplier. The supplier supplies all the products for the shipment and has its supply chain (irrelevant to reliance).

II Conceptual Data Modeling

1. EER Diagram

The EER diagram of the above database is shown in Figure 1 below to demonstrate the entities and their relations.



III Mapping Conceptual Model to Relational Model

Primary Key - Underlined

Foreign Key - *Italicized*

STORE (StoreID, AccNum, StoreAdd, StoreContact, Capacity)
EMPLOYEE (EmpID, StoreID, EmpName, EmpAdd, Salary, PhoneNum, Aadhar, DOB, PAN)
PRODUCT (ProdID, ProdName, MRP, Discount, Category, ExpiryDate, MFGDate, weight)
SUPPLIER (SupID, SupName, SupEmail, SupPhone, SupAddress)
CUSTOMER (CustID, CustName, CustPhone)
BILL (BillNum, CustID, Counter, PaymentMode, Date, BillAmount, TotalNumofItems)
ACCOUNT (AccNum, IFSC, BankName, TrnxDetails)
STORAGE_FACILITY (FacilityID, EmpID, FacilityName, FacilityAdd, PhoneNum)
RESTOCKING_SHIPMENT (RShipmentNum, FacilityID, StoreID)
SHIPMENT (ShipmentNum, SupID, FacilityID, Shipmentpayment)
PRODUCT_BELONGSTO_BILL (BillNum, ProdID, Qty)
CONTAINS (ProdID, ShipmentNum, QtyS)
CONTAINSR (ProdID, RShipmentNum, QtyRS)
SELL (ProdID, StoreID)
STORES (FacilityID, ProdID)

IV Implementation of Relational Model via MySQL and NoSQL

MYSQL Implementation:

The database was created in MYSQL and following queries were performed:

Query 1: Find the names of top-5 ranking products sold along with quantity. If there are multiple rows with the same quantity, display all.

Query:

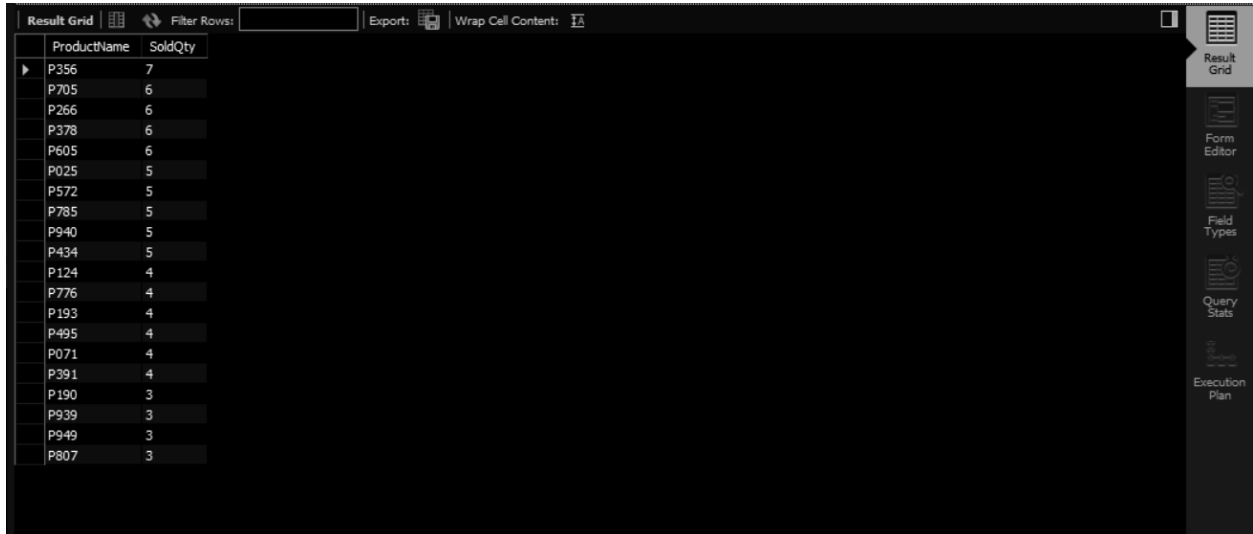
```
SELECT ProductName
       ,SoldQty
FROM (
    SELECT p.ProdName AS ProductName
           ,a.SoldQty AS SoldQty
           ,dense_rank() OVER (
               ORDER BY SoldQty DESC
           ) AS rk
    FROM (
        SELECT prodID
               ,count(*) AS SoldQty
        FROM product_belongsto_bill
        GROUP BY ProdID
    ) a
)
```

```

        JOIN product p ON a.ProdID = p.ProdID
    ) b
WHERE rk <= 5;

```

Output:



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays the following data:

ProductID	ProductName	SoldQty
P356		7
P705		6
P266		6
P378		6
P605		6
P025		5
P572		5
P785		5
P940		5
P434		5
P124		4
P776		4
P193		4
P495		4
P071		4
P391		4
P190		3
P939		3
P949		3
P807		3

The interface also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Contents' checkbox. On the right side, there is a vertical toolbar with icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'.

Query 2: 7Eleven wants to heavily invest in the top category sold across all stores. Write a query to display the name of the category along with the associated quantity.

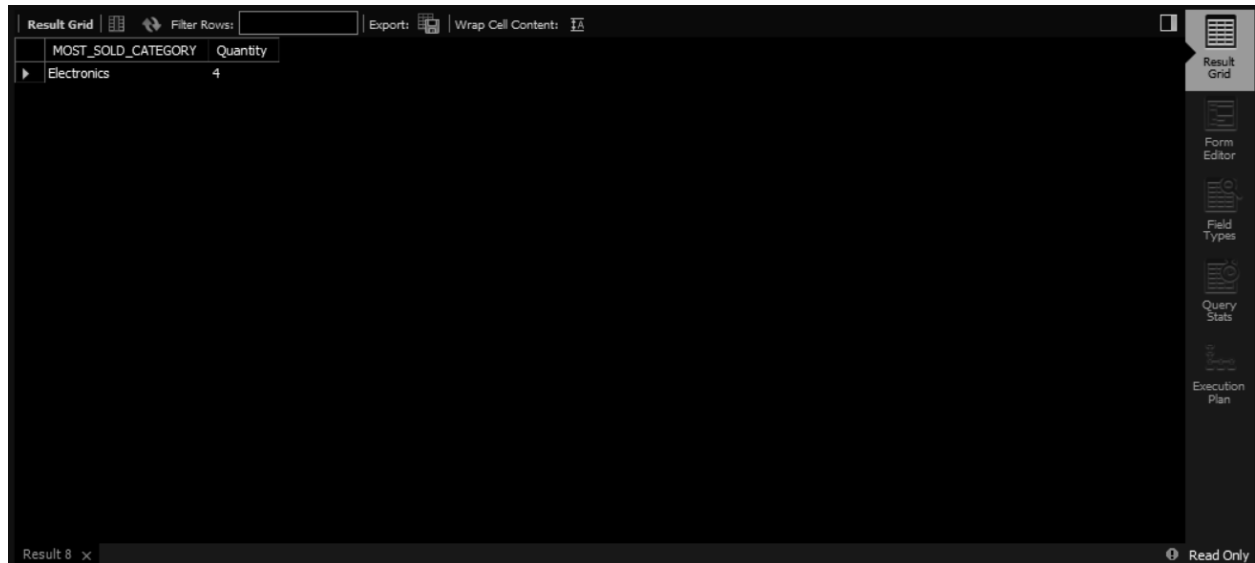
Query:

```

SELECT Category as MOST_SOLD_CATEGORY
       ,max(QtySold) AS Quantity
FROM (
    SELECT p.Category
           ,sum(pb.Qty) AS QtySold
    FROM product_belongsto_bill pb
    JOIN product p ON p.prodID = pb.ProdID
    GROUP BY Category
) a;

```

Output:



The screenshot shows a database query result grid. The top bar includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The grid has two columns: 'MOST_SOLD_CATEGORY' and 'Quantity'. A single row is displayed with 'Electronics' in the first column and '4' in the second column. The right sidebar contains icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'. The bottom status bar indicates 'Result 8' and 'Read Only'.

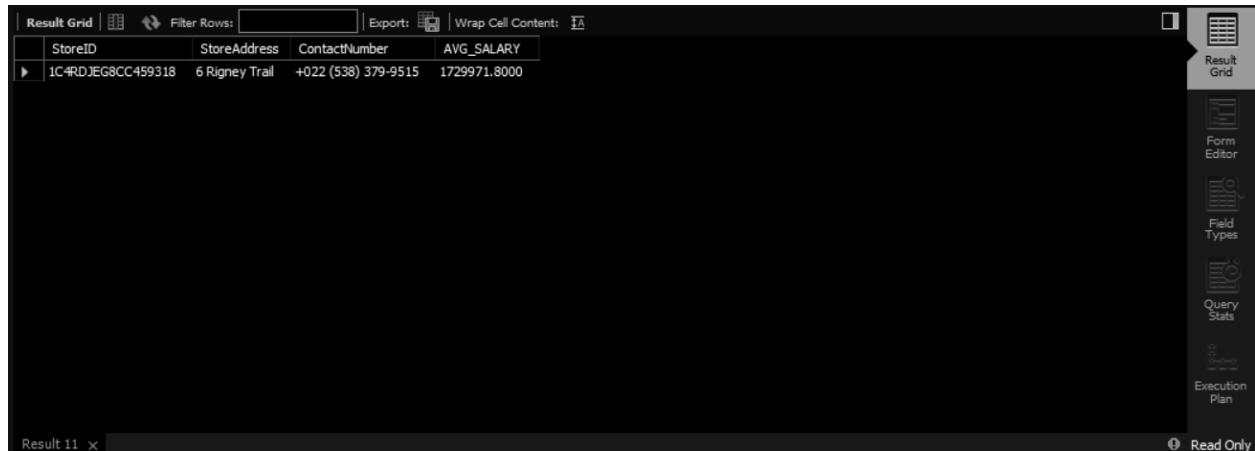
MOST_SOLD_CATEGORY	Quantity
Electronics	4

Query 3. 7Eleven wants to know the information of the store with the highest avg salary for employees.

Query:

```
SELECT e.s AS StoreID
      ,f.StoreAdd AS StoreAddress
      ,f.StoreContact AS ContactNumber
      ,max(a) AS AVG_SALARY
FROM (
      SELECT storeid AS s
            ,avg(salary) AS a
      FROM employee
      GROUP BY storeid
    ) e
JOIN store f ON f.storeid = e.s;
```

Output:



The screenshot shows a database query result grid with the following data:

StoreID	StoreAddress	ContactNumber	AVG_SALARY
1C4RDJEG8CC459318	6 Rigney Trail	+022 (538) 379-9515	1729971.8000

The interface includes a top toolbar with options like 'Filter Rows', 'Export', and 'Wrap Cell Content'. A right-hand sidebar contains icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'. The status bar at the bottom indicates 'Result 11' and 'Read Only'.

Query 4. In order to attract customers to shop regularly, 7Eleven had decided to target the customers who shop more and which is why the company launched a scheme called 'Customer of the month'. The customer of the month will get deals in the form of coupons which can be redeemed in the new 7-Eleven store. Display such customer information for each month.

```
WITH T
AS (
    SELECT month(DATE) AS Month
           ,c.CustD
           ,count(*) AS NumberOfVisits
    FROM customer c
    JOIN bill b ON b.CustD = c.custD
    GROUP BY Month
           ,CustD
)
SELECT Month
       ,C.CustD
       ,C.CustName
       ,NumberOfVisits
       ,C.PhoneNum
FROM T
JOIN Customer C ON C.CustD = T.CustD
WHERE (
    Month
    ,NumberOfVisits
) IN (
    SELECT Month
           ,max(NumberOfVisits)
    FROM T
    GROUP BY Month
);
```


Output:

Month	CustID	CustName	NumberOfVisits	PhoneNum
11	58-451-1365	Lyndsay Burcombe	6	(820) 5376080
12	98-191-7280	Garrick Rainer	4	(916) 7439605

Query 5. Display the most sold product month-wise for the supply-chain team

Query:

WITH T

AS (

SELECT ProdID

,month(DATE) AS Month

,sum(Qty) AS TotalQty

FROM product_belongsto_bill pb

JOIN bill b ON b.billNum = pb.billNum

GROUP BY ProdID

,Month

)

SELECT T.Month

,P.ProdID

,P.ProdName

FROM T

JOIN Product P ON T.ProdID = P.ProdID

WHERE (

Month

,TotalQty

) IN (

SELECT Month

,max(TotalQty)

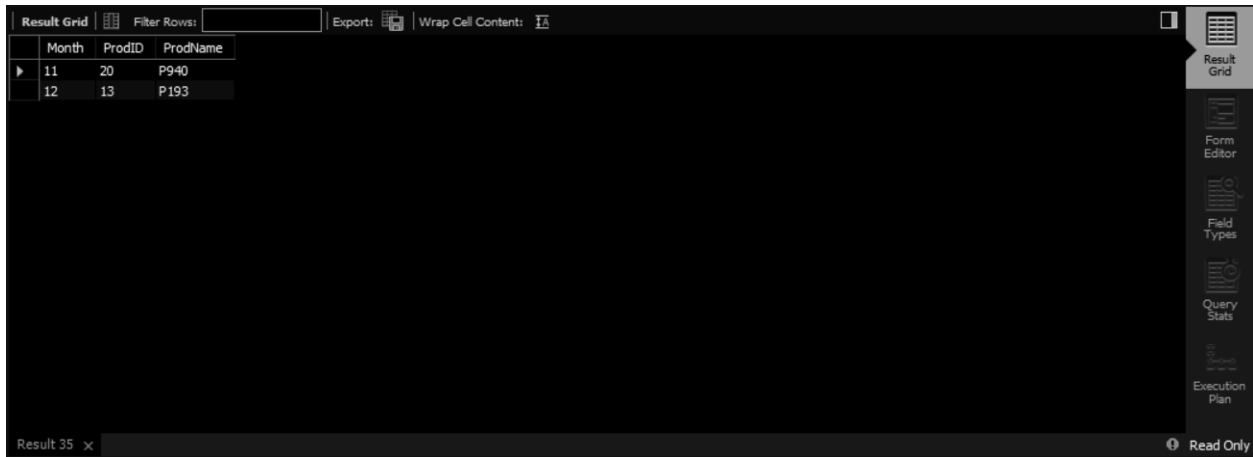
FROM T

GROUP BY Month

)

ORDER BY Month;

Output:



The screenshot shows a database query result grid. The grid has three columns: Month, ProdID, and ProdName. The first row shows Month 11, ProdID 20, and ProdName P940. The second row shows Month 12, ProdID 13, and ProdName P193. The grid is part of a larger application window with a sidebar on the right containing icons for Result Grid, Form Editor, Field Types, Query Stats, and Execution Plan. The bottom status bar indicates 'Result 35' and 'Read Only'.

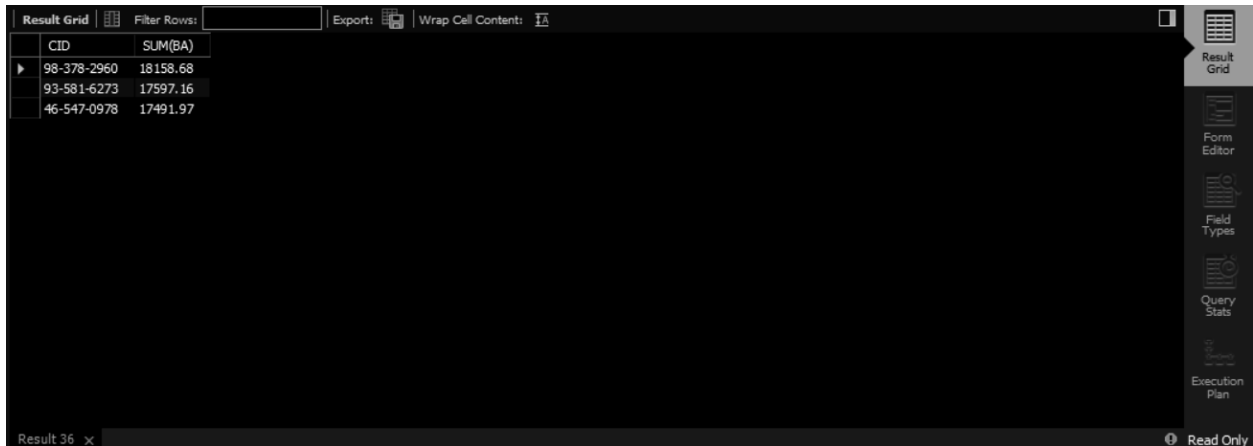
Month	ProdID	ProdName
11	20	P940
12	13	P193

Query 6: Write a query to display details of top 3 customers based on the total bill amount. Display CustomerID and Total Spend

Query

```
WITH BA (  
    BN  
    ,BA  
    ,CID  
)  
AS (  
    SELECT BILLNUM  
        ,CAST(REPLACE(REPLACE(BILLAMOUNT, ',', ''), '$', '' ) AS DECIMAL(10, 2))  
        ,CUSTD  
    FROM BILL  
)  
SELECT CID  
    ,SUM(BA)  
FROM BA  
GROUP BY CID  
ORDER BY SUM(BA) DESC limit 3;
```

Output:



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays three rows of data with columns 'CID' and 'SUM(BA)'. The first row is expanded, showing its details. The interface includes a top bar with 'Filter Rows', 'Export', and 'Wrap Cell Content' options. A right sidebar contains icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'. The bottom status bar indicates 'Result 36' and 'Read Only'.

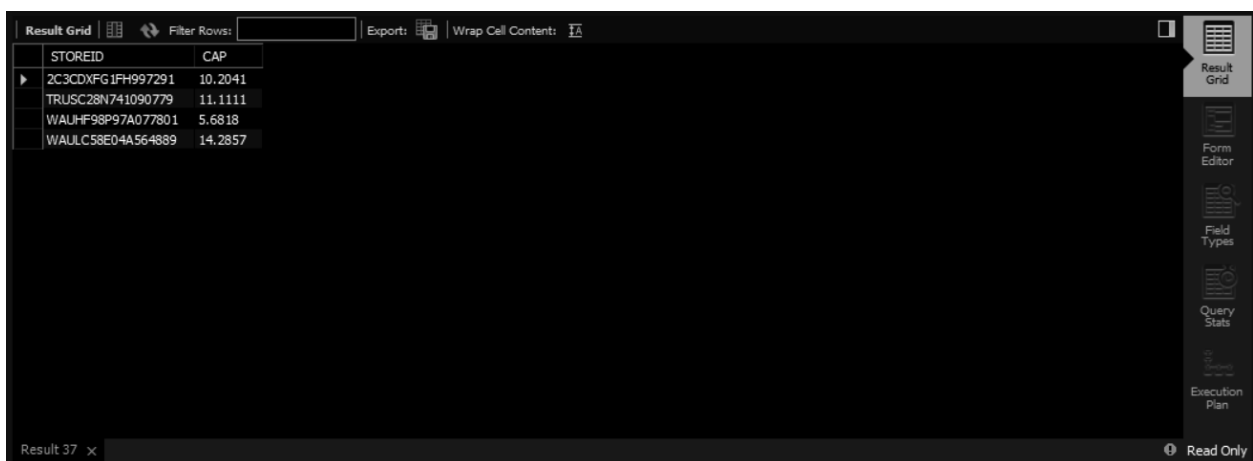
CID	SUM(BA)
98-378-2960	18158.68
93-581-6273	17597.16
46-547-0978	17491.97

Query 7: Government wants to regulate the number of EMPLOYEES who can be allowed in a store at a time due to Covid regulations and the company wants to share the analytics for the same, keeping a cap of 25% of the total store capacity. Show the stores that are following this regulation.

Query

```
SELECT storeid AS STOREID
       ,100 * count(e.Empld) / s.capacity AS CAP
FROM store s natural
JOIN employee e
GROUP BY s.storeid
HAVING CAP < 25
```

Output



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays four rows of data with columns 'STOREID' and 'CAP'. The first row is expanded, showing its details. The interface includes a top bar with 'Filter Rows', 'Export', and 'Wrap Cell Content' options. A right sidebar contains icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'. The bottom status bar indicates 'Result 37' and 'Read Only'.

STOREID	CAP
2C3CDXFG1FH997291	10.2041
TRUSC28N741090779	11.1111
WAUHF98P97A077801	5.6818
WAULC58E04A564889	14.2857

NoSQL Implementation

Two tables(Employee, Store) and one relation (Works_in) have been created in Neo4j playground. The following Cypher queries were done:

Query 1: Show the Store details along with the average income of the employees in each store.

```
MATCH(st:Store)-[e]-(emp:Employee)
RETURN st.StoreID as Store,st.StoreAdd as Address, st.StoreContact as contact,
avg(toInteger(emp.salary)) as Average_salary;
```

	Store	Address	contact	Average_salary
1	"1C4RDJEG8CC459318"	"6 Rigney Trail"	" +022 (538) 379-9515"	1369747.7857142861
2	"2C3CDXFG1FH997291"	"3 Fremont Plaza"	" +022 (993) 551-0840"	1167782.0000000002
3	"TRUSC28N741090779"	"68695 Russell Hill"	" +022 (379) 754-2326"	1512391.8
4	"WAUHF98P97A077801"	"2587 Bowman Trail"	" +022 (394) 155-4113"	1729971.8
5	"WAULC58E04A564889"	"2 Melvin Street"	" +022 (532) 220-8565"	1708459.909090909

Query 2: Show the Store details along with the number of employees in each store.

```
MATCH(st:Store)-[e]-(emp:Employee)
RETURN st.StoreID as Store,st.StoreAdd as Address, st.StoreContact as contact,
count(emp.EmpName) as Number_of_Employee;
```

	Store	Address	contact	Number_of_Employee
1	"1C4RDJEG8CC459318"	"6 Rigney Trail"	" +022 (538) 379-9515"	182
2	"2C3CDXFG1FH997291"	"3 Fremont Plaza"	" +022 (993) 551-0840"	130
3	"TRUSC28N741090779"	"68695 Russell Hill"	" +022 (379) 754-2326"	130
4	"WAUHF98P97A077801"	"2587 Bowman Trail"	" +022 (394) 155-4113"	65
5	"WAULC58E04A564889"	"2 Melvin Street"	" +022 (532) 220-8565"	143

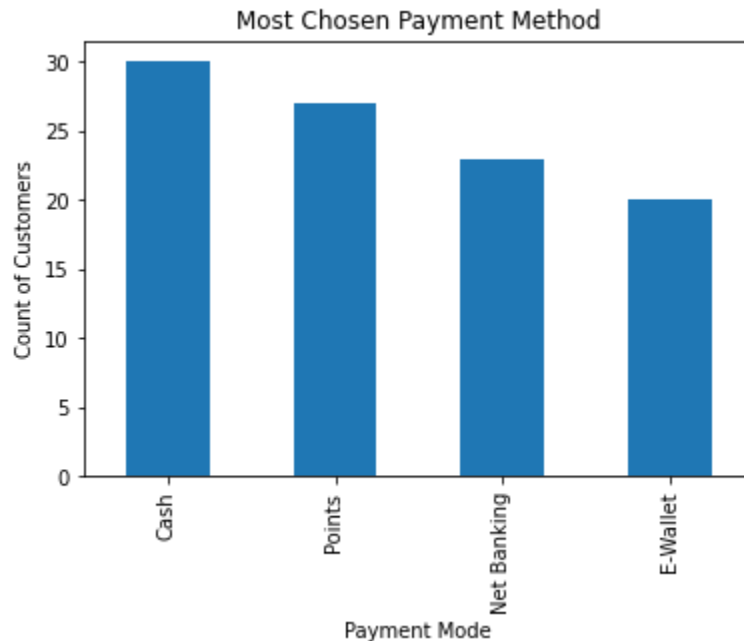
Query 3: Show all the Stores that have an average annual income more than 30000\$ (Rs.2250000) .

```
MATCH(st:Store)-[Cap]-(emp:Employee)
with st,emp,Cap,avg(toInteger(emp.salary)) as Avg
where Avg>2250000
return distinct st.StoreID
```

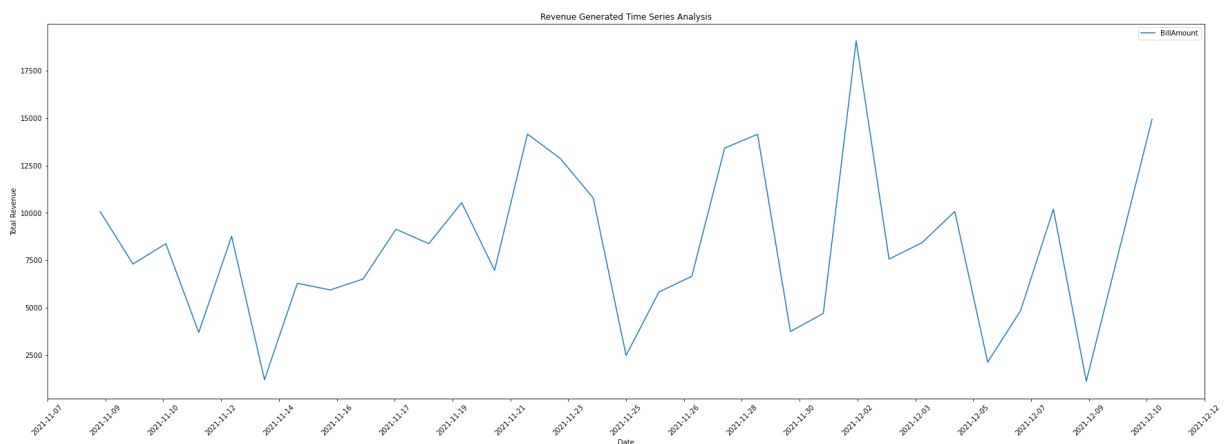
	st.StoreID
1	"TRUSC28N741090779"
2	"WAUHF98P97A077801"
3	"WAULC58E04A564889"

V. Database Access via Python

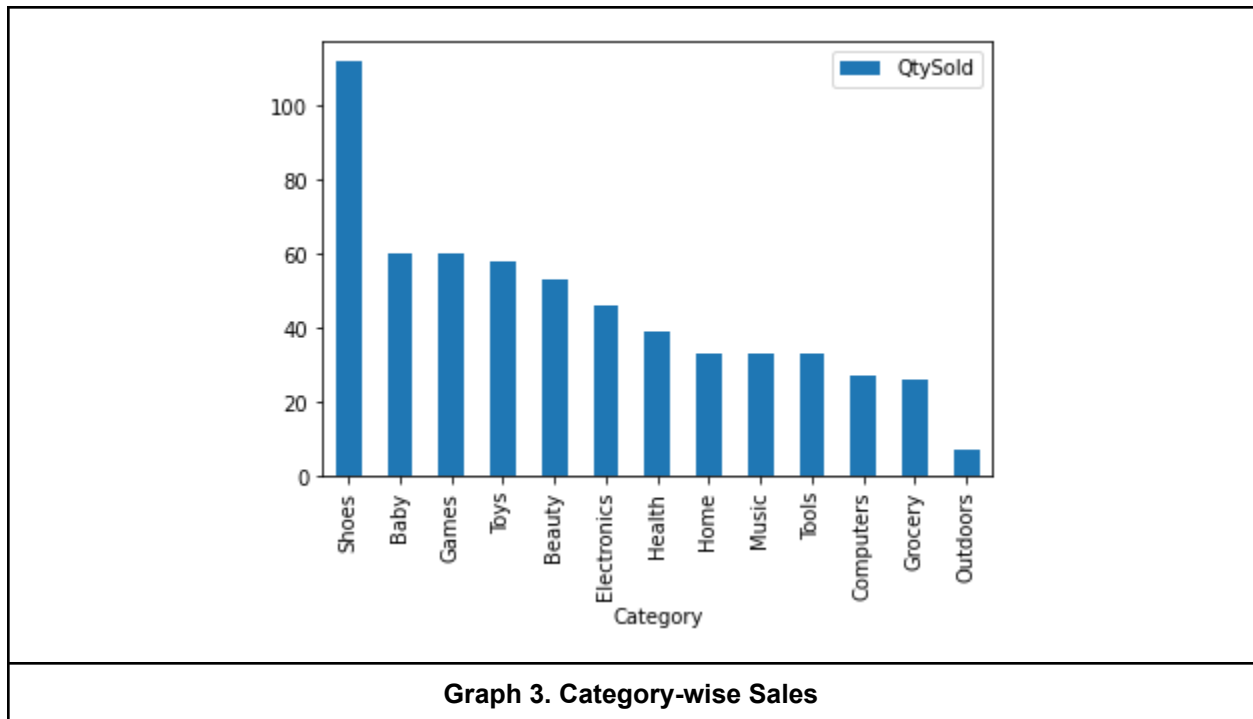
The database is accessed using Python to analyze and visualize data as shown below. The connection of MySQL to Python is established using MySQL.connector and cursor. execute is utilized to run the query and fetch results, followed by converting the list into a data frame using pandas library and using matplotlib to plot the graphs for the analytics.



Graph 1. Most Chosen Payment Method



Graph 2. Revenue Time Series Analysis



VI. Summary and Recommendation

The database access will facilitate 7-Eleven to establish itself firmly in the Indian market by leveraging the data and gaining result-driven insights. The database is designed and implemented on a MySQL database and can be directly used by 7-Eleven to access the data Reliance has decided to share.

The analytics generated via Python by establishing a connection with the MySQL database provides insights to answer some of the critical questions such as popular products. The analytics also gives them the stats related to marketing campaigns and the impact of incentives for both the customers and the employees. This will help 7-Eleven to understand the current market demand in an intuitive way and assess complex data with ease.

The improvements that can be further done are introducing the logistics part into the mechanism. The mode of transport, duration of delivery, and expected delivery dates can be added to the database. This will give a better understanding of how the logistics aspect works in India as well as how penalties are given based on the number of days by which the delivery is delayed.