Coffee Show UNFC: Database Design and Implementation

Abstract

This project on database design and implementation stems from the idea of representing coffee as a central part of our daily lives as students. By focusing on a business that seeks growth, we aim to explore how data can be leveraged for strategic decision-making. Our goal is to understand the business context in which a company operates and how detailed data analysis can drive its growth. To achieve this, we will design a comprehensive SQL database capable of handling data definition statements (DDL), data manipulation statements (DML), and data retrieval statements (DQL). This structured approach will enable efficient management and analysis of data, supporting the company's growth objectives through informed decision-making processes.

Contents

1. Introduction

Chiu et al. (2012) emphasized that finding factors influencing brand loyalty is becoming more important in an increasingly competitive business market. To effectively analyze customer behavior and make informed decisions, it is essential to understand historical data such as transaction records, sales trends, spending patterns, purchased products, and frequently visited stores. All this information must be stored and processed within a robust dataset system to enable confident data-driven decision-making based on historical insights.

This project focuses on the design, implementation, and analysis of a relational SQL database tailored for a coffee shop, emphasizing data integrity, scalability, and usability. By incorporating key relationships between entities such as products, categories, customers, employees, transactions, rewards, and store locations, the database reduces data duplication and optimizes data processing efficiency.

The report contains the process of developing the database, including the schema design, the use of SQL statements for data definition and manipulation, and the use of robust queries to analyze behavior, trends, and patterns. The result is a comprehensive and well-connected system that stores information and can be used for the operations and analytical needs of a coffee shop.

2. Objectives

- Design a normalized database for a coffee shop business.
- Demonstrate our ability to manage the database efficiently and professionally.

- Understand the customer behavior by group segmentation.
- Understand seasonal patterns of purchases by store, tailoring promotions, discounts, and rewards for customers.
- Understand demographic differences and behavior among customers. For example genre and age.

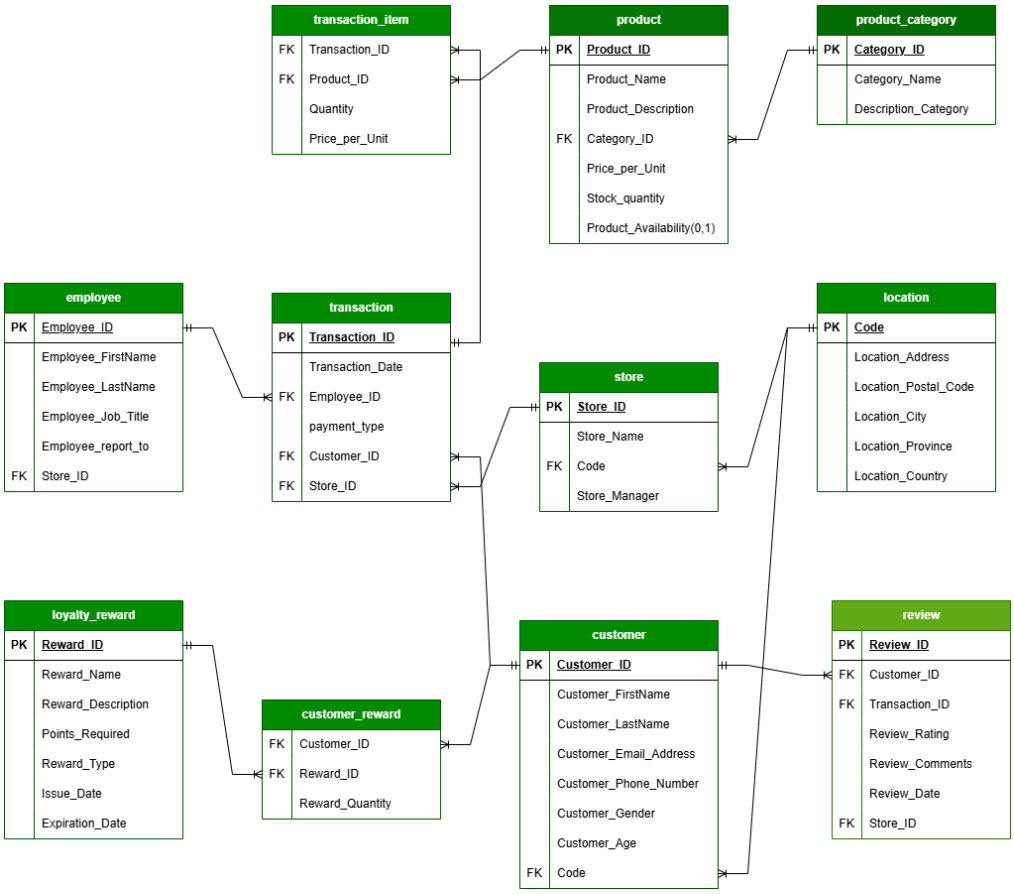
3. Database Design

The database was designed to follow the main transactions of a coffee shop, it records information about products, transactions, stores, customers, and reviews. It contains eleven tables and follows the principles of normalization to ensure data integrity and eliminate redundancy. It follows all the steps:

- Definition of strong entities,
- Entity attributes,
- Relationships,
- Cardinality ratio,
- Participation constraints,
- Definition of weak entities and weak relationships.

3.1. Entity-Relationship Diagram:

The ERD illustrates the structure of the database with 11 interrelated tables. It models key entities such as products, transactions, customers, employees, and rewards. Primary and foreign keys are implemented to ensure data integrity, while the schema design promotes clean and efficient data retrieval, which is essential for both reporting and analysis.



3.2. Database Dictionary

Table: product_category

Field Name	Data Type	Size	Description
Category_ID(PK)	INT	N/A	Unique identifier for the product category
Category_Name	VARCHAR(50)	50	Name of the product category
Description_Category	VARCHAR(255)	255	Detailed description of the product category

Table: product

Field Name	Data Type	Size	Description
Product_ID(PK)	INT	N/A	Unique identifier for the product
Product_Name	VARCHAR(50)	50	Name of the product
Product_Description	VARCHAR(255)	255	Detailed description of the product
Category_ID(FK)	INT	N/A	Reference to the category of the product
Price_per_Unit	DECIMAL(10,2)	10,2	Price per unit of the product
Stock_quantity	INT	N/A	Number of units available in stock
Product_Availability	TINYINT(1)	1	Availability status of the product (1 for available, 0 for unavailable)

Table: store

Field Name	Data Type	Size	Description
Store_ID(PK)	INT	N/A	Unique identifier for the store
Store_Name	VARCHAR(50)	50	Name of the store
Code	VARCHAR(50)	50	Unique store code
Store_Manager	VARCHAR(255)	255	Name of the store manager

Table: employee

Field Name	Data Type	Size	Description
Employee_ID(PK)	INT	N/A	Unique identifier for the employee
Employee_FirstName	VARCHAR(50)	50	First name of the employee
Employee_LastName	VARCHAR(50)	50	Last name of the employee
Employee_Job_Title	VARCHAR(100)	100	Job title of the employee
Employee_Report_To	INT	N/A	The ID of the manager the employee reports to
Store_ID(FK)	INT	N/A	Reference to the store the employee is assigned to

Table: location

Field Name	Data Type	Size	Description
Code(PK)	VARCHAR(50)	50	Unique location code
Location_Address	VARCHAR(255)	255	Address of the location
Location_Postal_Code	VARCHAR(6)	6	Postal code of the location
Location_City	VARCHAR(50)	50	The city where the location is situated
Location_Province	VARCHAR(50)	50	The province where the location is situated
Location_Country	VARCHAR(50)	50	The country where the location is situated

Table: customer

Field Name	Data Type	Size	Description
Customer_ID(PK)	INT	N/A	Unique identifier for the customer
Customer_FirstName	VARCHAR(255)	255	First name of the customer
Customer_LastName	VARCHAR(255)	255	Last name of the customer
Customer_Email_Address	VARCHAR(255)	255	Unique email address of the customer
Customer_Phone_Number	VARCHAR(20)	20	Phone number of the customer
Customer_Gender	ENUM('Male', 'Female', 'Other')	N/A	Gender of the customer (Male, Female, Other)
Customer_Age	INT	N/A	Age of the customer (must be non-negative)
Code(FK)	VARCHAR(50)	50	Location code referencing the location table

Table: transaction

Field Name	Data Type	Size	Description
Transaction_ID(PK)	INT	N/A	Unique identifier for the transaction
Transaction_Date	DATETIME	N/A	Date and time of the transaction
Employee_ID(FK)	INT	N/A	An employee who handled the transaction
Payment_Type	VARCHAR(50)	50	Type of payment method used

Customer_ID(FK)	INT	N/A	The customer involved in the transaction
Store_ID(FK)	INT	N/A	Store where the transaction took place

Table: transaction_item

Field Name	Data Type	Size	Description
Transaction_ID(PK,FK)	INT	N/A	Identifier for the transaction
Product_ID(PK,FK)	INT	N/A	Identifier for the purchased product
Quantity	INT	N/A	Quantity of the product in the transaction
Price_per_Unit	DECIMAL(10,2)	10,2	Price per unit of the product

Table: review

Field Name	Data Type	Size	Description
Review_ID(PK)	INT	N/A	Unique identifier for the review
Customer_ID(FK)	INT	N/A	identifier for the customer
Transaction_ID(Fk)	INT	N/A	Transaction associated with the review
Store_ID(FK)	INT	N/A	Store associated with the review
Review_Rating	INT	N/A	Rating given by the customer (1 to 5)
Review_Comments	VARCHAR(255)	255	Customer comments on the review
Review_Date	DATETIME	N/A	Date and time the review was created

Table: loyalty_reward

Field Name	Data Type	Size	Description
Reward_ID(PK)	INT	N/A	Unique identifier for the reward
Reward_Name	VARCHAR(255)	255	Name of the reward
Reward_Description	TEXT	N/A	Description of the reward
Points_Required	INT	N/A	Points required to redeem the reward
Reward_Type	VARCHAR(100)	100	Type of reward (e.g., discount, free item)
Issue_Date	DATE	N/A	Date the reward was issued
Expiration_Date	DATE	N/A	Date the reward expires

Table: customer_reward

Field Name	Data Type	Size	Description
Customer_ID(PK,FK)	INT	N/A	Unique identifier for the customer
Reward_ID(PK,FK)	INT	N/A	Unique identifier for the reward
Reward_Quantity	INT	N/A	Quantity of the reward redeemed by the customer

3.3. Relationships

- product_category vs product: 1:M
- product vs transaction_item: 1:M
- transaction_items vs transaction: M:1
- employee vs transaction: 1:M
- transaction vs store M:1
- store vs location M:1
- review vs transaction 1:1
- customer vs review 1:M
- customer vs location M:1
- store vs location M:1
- customer vs customer_reward: 1:M
- customer_reward vs loyalty_reward M:1
- employee vs store M:1

4. Data Definition Statements (DDL)

Using SQL Data Definition Language (DDL) statements, the database tables and their relationships were created as listed below:

4.1. Database Creation

The database named coffeeshowunfc will store all the relevant information necessary to understand the business.

```
#CREATE DATABASE coffeeshowunfc *.*
CREATE DATABASE coffeeshowunfc;
USE coffeeshowunfc;
```

4.2. Table Creation

The following queries demonstrate Data Definition Language DDL for table creation and the imposition of constraints, such as referential integrity. Specified the attributes, types, and constraints of a table, including primary keys, and foreign keys of this table along with their referenced primary keys in other tables.

```
#CREATE TABLE coffeeshowunfc *.*
CREATE TABLE coffeeshowunfc.product_category (
    Category_ID INT PRIMARY KEY NOT NULL,
    Category_Name VARCHAR(50) NOT NULL,
    Description_Category VARCHAR(255)
);
```

```
CREATE TABLE coffeeshowunfc.product (
    Product_ID INT PRIMARY KEY NOT NULL,
    Product_Name VARCHAR(50) NOT NULL,
    Product_Description VARCHAR(255),
    Category_ID INT,
    Price_per_Unit DECIMAL(10,2) NOT NULL,
    Stock_quantity INT NOT NULL,
    Product_Availability TINYINT(1) NOT NULL DEFAULT 1,
    FOREIGN KEY (Category_ID) REFERENCES product_category(Category_ID) ON DELETE SET NULL
);
```

```sql
CREATE TABLE coffeeshowunfc.store (
    Store_ID INT PRIMARY KEY NOT NULL,
    Store_Name VARCHAR(50) NOT NULL,
    Code VARCHAR(50) UNIQUE,
    Store_Manager VARCHAR(255)
);
```

```sql
CREATE TABLE coffeeshowunfc.employee (
    Employee_ID INT PRIMARY KEY NOT NULL,
    Employee_FirstName VARCHAR(50) NOT NULL,
    Employee_LastName VARCHAR(50) NOT NULL,
    Employee_Job_Title VARCHAR(100) NOT NULL,
    Employee_Report_To INT,
    Store_ID INT,
    FOREIGN KEY (Store_ID) REFERENCES store(Store_ID) ON DELETE SET NULL
);
```

```sql
CREATE TABLE coffeeshowunfc.location (
    Code VARCHAR(50) PRIMARY KEY,
    Location_Address VARCHAR(255),
    Location_Postal_Code VARCHAR(6),
    Location_City VARCHAR(50),
    Location_Province VARCHAR(50),
    Location_Country VARCHAR(50)
);
```

```sql
CREATE TABLE coffeeshowunfc.customer (
    Customer_ID INT PRIMARY KEY NOT NULL,
    Customer_FirstName VARCHAR(255) NOT NULL,
    Customer_LastName VARCHAR(255) NOT NULL,
    Customer_Email_Address VARCHAR(255) UNIQUE NOT NULL,
    Customer_Phone_Number VARCHAR(20),
    Customer_Gender ENUM('Male', 'Female', 'Other'),
    Customer_Age INT CHECK (Customer_Age >= 0),
    Code VARCHAR(50),
    FOREIGN KEY (Code) REFERENCES location(Code) ON DELETE SET NULL
);
```

```sql
CREATE TABLE coffeeshowunfc.transaction (
    Transaction_ID INT PRIMARY KEY NOT NULL,
    Transaction_Date DATETIME DEFAULT CURRENT_TIMESTAMP,
    Employee_ID INT,
    Payment_Type VARCHAR(50),
    Customer_ID INT,
    Store_ID INT,
    FOREIGN KEY (Employee_ID) REFERENCES employee(Employee_ID) ON DELETE SET NULL,
    FOREIGN KEY (Customer_ID) REFERENCES customer(Customer_ID) ON DELETE CASCADE,
    FOREIGN KEY (Store_ID) REFERENCES store(Store_ID) ON DELETE SET NULL
);
```

```sql
CREATE TABLE coffeeshowunfc.transaction_item (
    Transaction_ID INT NOT NULL,
    Product_ID INT NOT NULL,
    Quantity INT NOT NULL,
    Price_per_Unit DECIMAL(10,2) NOT NULL,
    PRIMARY KEY (Transaction_ID, Product_ID),
    FOREIGN KEY (Transaction_ID) REFERENCES transaction(Transaction_ID) ON DELETE CASCADE,
    FOREIGN KEY (Product_ID) REFERENCES product(Product_ID) ON DELETE CASCADE
);
```

```sql
CREATE TABLE coffeeshowunfc.review (
    Review_ID INT PRIMARY KEY NOT NULL,
    Customer_ID INT,
    Transaction_ID INT NULL,
    Store_ID INT NULL,
    Review_Rating INT CHECK (Review_Rating BETWEEN 1 AND 5),
    Review_Comments VARCHAR(255),
    Review_Date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (Customer_ID) REFERENCES customer(Customer_ID) ON DELETE CASCADE,
    FOREIGN KEY (Transaction_ID) REFERENCES transaction(Transaction_ID) ON DELETE SET NULL,
    FOREIGN KEY (Store_ID) REFERENCES store(Store_ID) ON DELETE SET NULL
);
```

```sql
CREATE TABLE coffeeshowunfc.loyalty_reward (
    Reward_ID INT PRIMARY KEY NOT NULL,
    Reward_Name VARCHAR(255) NOT NULL,
    Reward_Description TEXT,
    Points_Required INT NOT NULL,
    Reward_Type VARCHAR(100),
    Issue_Date DATE,
    Expiration_Date DATE
);
```

```sql
CREATE TABLE coffeeshowunfc.customer_reward (
    Customer_ID INT NOT NULL,
    Reward_ID INT NOT NULL,
    Reward_Quantity INT NOT NULL DEFAULT 1,
    PRIMARY KEY (Customer_ID, Reward_ID),
    FOREIGN KEY (Customer_ID) REFERENCES customer(Customer_ID) ON DELETE CASCADE,
    FOREIGN KEY (Reward_ID) REFERENCES loyalty_reward(Reward_ID) ON DELETE CASCADE
);
```

5. Data Manipulation Statements (DML)

5.1. Population of tables

Initially, tables were created using dummy values to evaluate the quality and performance of the database. Then, with AI tools, the data was collected using a specific and robust

prompt with the relations and type of data previously defined, allowing the creation of real and cohesive information without bias.

The following code shows an example of how the data was imported and uploaded into the database allowing further steps. To review the full data insert, open "Group7_Project_CoffeeShowUNFC_Final.sql" in the section "2. DML Final Project CoffeeShowUNFC -- INSERT Values for each table --"

```
INSERT INTO `employee`
VALUES
(1,'Melanie','Rios','Manager',1,1),
(2,'Robert','Greene','Cashier',1,1),
(3,'John','Parker','Manager',1,2),
(4,'David','Richardson','Barista',1,2),
(5,'Jeremy','Russell','Cashier',1,3),
(6,'Victoria','Ray','Manager',1,3),
(7,'Kathy','Williams','Manager',1,4),
(8,'Valerie','Larson','Manager',1,4),
(9,'Joshua','Bennett','Barista',1,5),
(10,'Donald','Lin','Cashier',1,5),
(11,'Jeff','Scott','Cashier',1,6),
(12,'Jeremy','Vazquez','Cashier',1,6);
```

```
INSERT INTO `store`
    VALUES
        (1,'Coffee Haven Ontario','93','Zachary Davis'),
        (2,'Coffee Haven Quebec','75','Mary Moyer'),
        (3,'Coffee Haven British Columbia','39','Luis Mills'),
        (4,'Coffee Haven Alberta','6','Andrea Barnett'),
        (5,'Coffee Haven Manitoba','20','Patrick Webb'),
        (6,'Coffee Haven Nova Scotia','13','Yolanda Duncan');
```

5.2. Database Operations for Launching Daily Transaction

Adding new information to the database system is part of the daily operations in a coffee shop and plays a key role in supporting the company's growth. For instance, every time a new customer walks in to purchase something, a new transaction is created and recorded in the system.

To create the transaction first, the employee needs to create the customer, so he asks for their personal information such as name, last name, mail, phone number, gender, age, and location to insert into the database.

```
INSERT INTO customer
VALUES ('101','Juan','Cisneros','','416-8222-981','Male','23','80');
```

And finally, the customer orders a cup of coffee and pays with a debit card:

```
INSERT INTO transaction
VALUES ('36501','2024-09-10','2','Debit Card','101','1');
INSERT INTO transaction_item
VALUES ('36501','1',1,8.11);
```

Regarding with company's growth, insert values can be useful if the company wants to open a new store.

```
-- INSERT --
INSERT INTO `location` VALUES ('101','12012 Avenue West','S6B5C1','Prince Albert','Saskatchewan','Canada');
INSERT INTO `store` VALUES (7,'Coffee Haven Saskatchewan','101','James Bond')
```

5.3. Database Operations for Updating Customer Personal Information

Update customer information such as email, address, and phone number.

```
-- UPDATE --
UPDATE coffeeshowunfc.customer SET Customer_Phone_Number='4354448691'
where Customer_ID=100;
```

5.4. Database Operations for Removing a Closing Store

In some cases when the fixed costs are higher than the profit, a store needs to be relocated or even closed. In this case, deleting a row is important.

```
DELETE FROM  coffeeshowunfc.store
WHERE store_id=7;
```

6. Data Retrieval Statements (DQL)

Provides data retrieval and analysis tools, including aggregation functions and statistical summaries to explain the data.

6.1. View of sales per item

The company can gain clearer insights into earnings per product by creating a view that provides a better understanding of product performance and sales trends.

```
CREATE VIEW coffeeshowunfc.sales_item_transaction AS
SELECT
Transaction_ID,Product_ID,Quantity, Price_per_Unit,
COALESCE(quantity*price_per_unit,0) AS sales_per_item
FROM coffeeshowunfc.transaction_item;
```

6.2. View of sales per transaction

By creating a view, the company can visualize the average spending per customer per purchase, gaining more accurate insights into purchasing behavior for both the company and its customers.

```
CREATE VIEW coffeeshowunfc.sales_transaction AS
SELECT
Transaction_ID,
SUM(COALESCE(quantity*price_per_unit,0)) AS sales_per_transaction
FROM coffeeshowunfc.transaction_item
GROUP BY Transaction_ID;
```

6.3. View of accumulated loyalty points

Both employees and customers can easily track the total accumulated points. Additionally, this view helps the company measure the success of the loyalty program by observing if purchase patterns improve as customers earn and redeem points.

```
create or replace view coffeeshowunfc.acum_loyalty_points as(
    select
        cus.Customer_ID as Customer_ID,
        cus.Customer_FirstName,
        cus.Customer_LastName,
        cus.Customer_Email_Address,
        cus.Customer_Phone_Number,
        cus.Customer_Gender,
        cus.Customer_Age,
        tr.Transaction_ID,
        tr.Transaction_Date as Transaction_Date,
        tri.Product_ID,
        tri.Quantity,
        tri.Price_per_Unit,
        ROUND(((tri.Quantity*tri.Price_per_Unit)*0.1),2) as  acum_loyalty_points
    from coffeeshowunfc.customer cus
        JOIN coffeeshowunfc.transaction tr ON cus.customer_id=tr.customer_id
        JOIN coffeeshowunfc.transaction_item tri ON tr.transaction_id=tri.transaction_id
    Order by Transaction_Date desc
);
```

6.4. Customer Behavior Analysis

Group customers into four spending categories to identify high-value customers and the ones that need incentives to increase purchases. By looking at demographics and spending habits, the company can create targeted promotions for each group. Track how often customers buy to send reminders to those with longer gaps and offer exclusive deals to loyal shoppers. Additionally, identifying top stores for each customer allows the company

to examine if there are patterns between high-spending or frequent-visit groups and specific locations, helping to optimize store-specific marketing strategies and improve retention.

6.4.1. Query

The query utilizes CTEs, aggregation, and window functions for its development.

```
WITH
customer_behavior AS
            (SELECT
                t.customer_ID, t.transaction_date AS current_purchase,
                LAG(t.transaction_date) OVER(PARTITION BY t.customer_ID ORDER BY t.transaction_date)
                AS previous_purchase,
                DATEDIFF(t.transaction_date, LAG(t.transaction_date)
                OVER(PARTITION BY t.customer_ID ORDER BY t.transaction_date)) AS days_between_orders
            FROM coffeeshowunfc.transaction t),
average_total AS
            (SELECT
                AVG(days_between_orders) AS total_average
            FROM customer_behavior),


classified_customers AS (
            SELECT
                customer_id, AVG(days_between_orders) AS avg_days_between_orders,
                (SELECT total_average FROM average_total) AS mean,
                CASE
                WHEN AVG(days_between_orders)<(SELECT total_average FROM average_total) THEN 'Less days than average'
                WHEN AVG(days_between_orders)=(SELECT total_average FROM average_total) THEN 'Average'
                WHEN AVG(days_between_orders)>(SELECT total_average FROM average_total) THEN 'More days than average'
                END AS classification
            FROM customer_behavior
            GROUP BY customer_id
            ORDER BY avg_days_between_orders),


total_per_cat AS (
                SELECT
                    classification, count(*) AS total_customers_per_category
                FROM classified_customers
                GROUP BY classification),


customer_behavior_classifications AS (
                SELECT
                    cc.customer_id, cc.avg_days_between_orders, cc.classification,
                    tpc.total_customers_per_category
                FROM classified_customers cc
                    JOIN total_per_cat tpc ON tpc.classification=cc.classification),
```

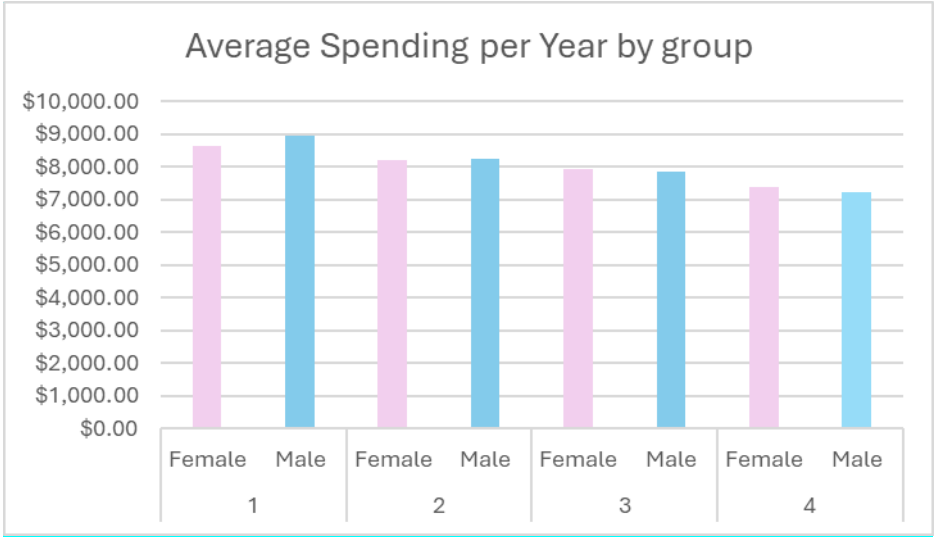
```
Ranked_Stores AS (
            SELECT
                customer_id, store_id,
                COUNT(*) AS transaction_count,
                RANK() OVER (PARTITION BY customer_id ORDER BY COUNT(*) DESC) AS ranking
            FROM coffeeshowunfc.transaction
            GROUP BY customer_id, store_id
            ),
top_stores_per_client AS (
            SELECT
                customer_id, store_id
            FROM Ranked_Stores
            WHERE ranking = 1)

SELECT
    c.customer_ID, c.customer_FirstName, c.customer_LastName, c.customer_gender, c.customer_age,
    COALESCE(SUM(ti.quantity*ti.price_per_unit),0) AS total_amount_spent,
    NTILE(4) OVER(ORDER BY COALESCE(SUM(ti.quantity*ti.price_per_unit),0) DESC) AS purchasing_group,
    cbc.avg_days_between_orders, cbc.classification, cbc.total_customers_per_category,
    tspc.store_id
FROM coffeeshowunfc.customer c
    LEFT JOIN coffeeshowunfc.transaction t ON c.customer_ID = t.customer_ID
    LEFT JOIN coffeeshowunfc.transaction_item ti ON ti.Transaction_ID = t.Transaction_ID
    LEFT JOIN customer_behavior_classifications cbc ON cbc.customer_ID = c.customer_ID
    LEFT JOIN top_stores_per_client tspc ON tspc.customer_ID = c.customer_ID
GROUP BY c.customer_ID, cbc.avg_days_between_orders, cbc.classification,
cbc.total_customers_per_category, tspc.store_id
ORDER BY total_amount_spent DESC;
```

6.4.2. Results

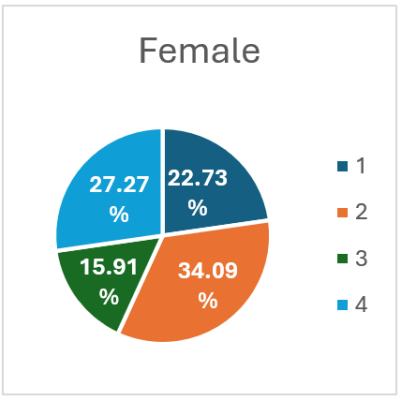
Customers were grouped into four categories where “1” represents the high-spending, “2” moderate-high-spending, “3” moderate-spending, and “4” low-spending.

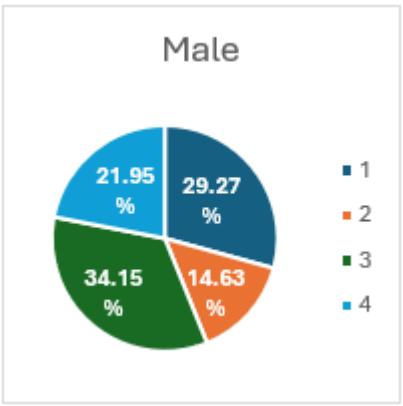
Average Total Amount Spent	Female	Male
1	$ 8,657.72	$ 8,960.11
2	$ 8,210.50	$ 8,230.37
3	$ 7,948.88	$ 7,837.98
4	$ 7,369.53	$ 7,243.10



Based on this information, the company can see that, despite categorizing customers by yearly spending amounts, there is little difference between the first three groups, with less than $1000 separating them. Additionally, male customers in the highest spending groups tend to spend slightly more, while female customers in the lowest devote a bit more. However, overall, there is no significant difference between the groups by gender, suggesting that marketing strategies should not be gender-based to boost purchases, as the behaviors are quite similar.

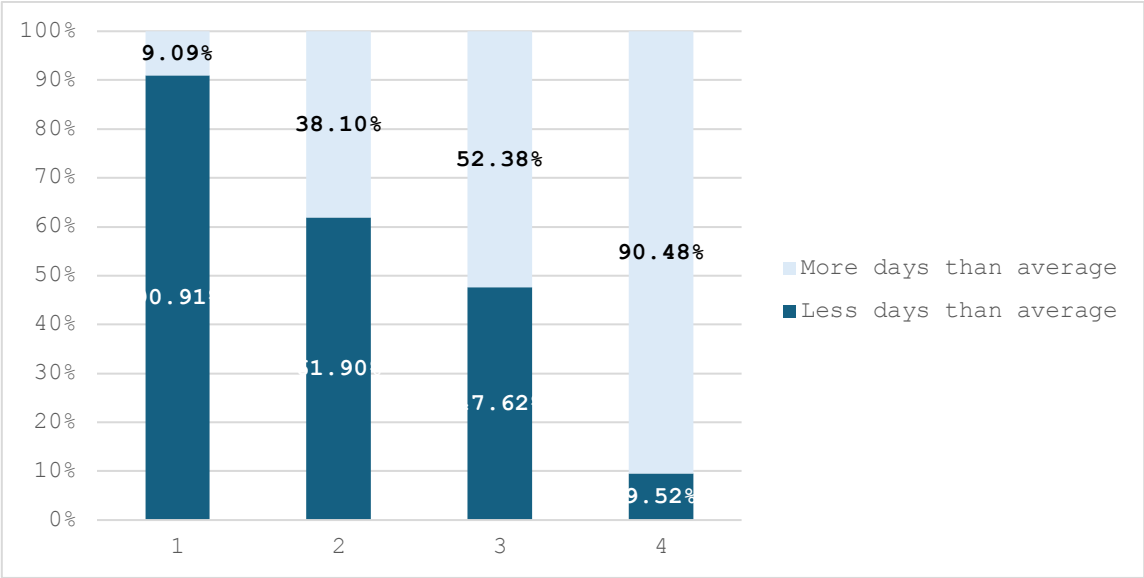
Count of purchasing_group	Female	Male	Grand Total
1	22.73%	29.27%	25.88%
2	34.09%	14.63%	24.71%
3	15.91%	34.15%	24.71%
4	27.27%	21.95%	24.71%





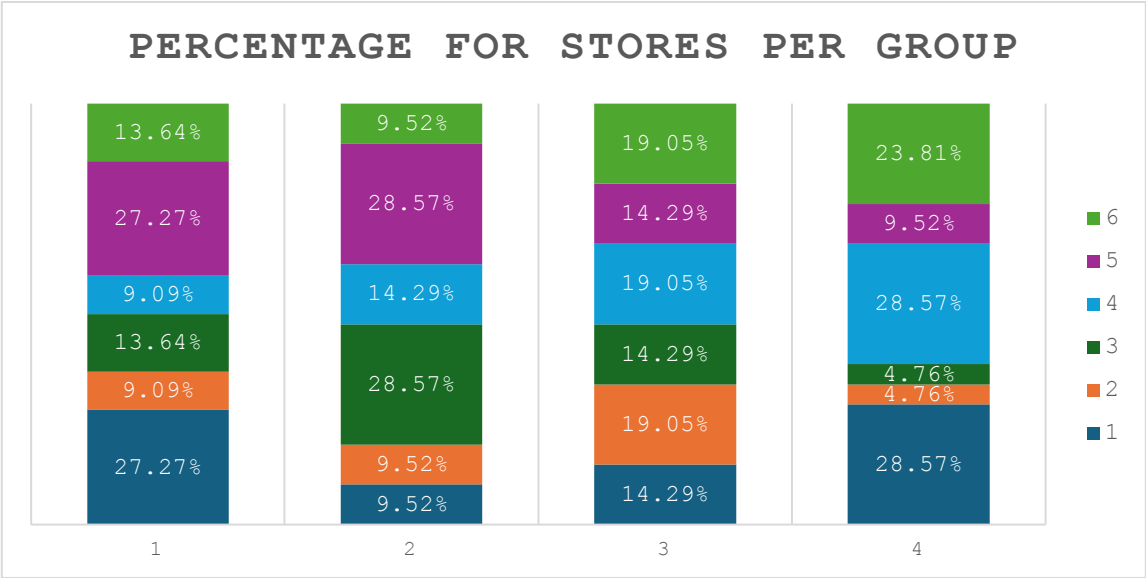
Considering this data, it is evident that female and male customers exhibit similar behavior in groups 1 and 4. However, groups 3 and 4 appear to be somewhat inverted: most males belong to group 3 (34.15%), while group 2 has the fewest males (14.63%). For females, the pattern is the opposite, group 2 has the highest percentage of females (34.09%), and group 3 has the lowest (15.91%). This suggests that while there are some differences in group distribution between genders, the overall purchasing behavior does not vary drastically, this reaffirms that gender-based segmentation may not be the most effective strategy for targeting marketing efforts.

Count of classification	Less days than	More days than average	Grand Total
1	90.91%	9.09%	100.00%
2	61.90%	38.10%	100.00%
3	47.62%	52.38%	100.00%
4	9.52%	90.48%	100.00%
Grand Total	**52.94%**	**47.06%**	**100.00%**



When analyzing this information, a clear pattern is observed: the highest purchasing group tends to have the shortest time between orders, while the lowest purchasing group has the most customers who make purchases less frequently. The trend suggests that as spending decreases, fewer customers visit the store more than the average amount. This provides valuable insight that average spending is closely linked to purchasing frequency since customers tend to make similar-sized purchases on average and the categorization of each group seems to be heavily influenced by them. Therefore, the key strategy should focus on increasing visit frequency, potentially through offers, promotions, or other incentives, to drive higher profits.

PURCHASING GROUP OF STORE	1	2	3	4	5	6	Grand Total
1	27.27%	9.09%	13.64%	9.09%	27.27%	13.64%	100.00%
2	9.52%	9.52%	28.57%	14.29%	28.57%	9.52%	100.00%
3	14.29%	19.05%	14.29%	19.05%	14.29%	19.05%	100.00%
4	28.57%	4.76%	4.76%	28.57%	9.52%	23.81%	100.00%
Grand Total	**20.00%**	**10.59%**	**15.29%**	**0.1765**	**20.00%**	**16.47%**	**100.00%**



Analyzing customer preferences across purchasing groups reveals that stores 1 and 5 are the most favored, with store 5 particularly popular among high-spending customers. To capitalize on this, the company should examine the successful strategies employed by these stores, especially store 5, and consider replicating them across other locations. Additionally, conducting thorough analyses of less-visited stores, including gathering customer feedback, assessing location factors, and evaluating other relevant variables, can provide insights for targeted improvements.

6.5. Store Analysis

Analyze store sales performance to identify high-performing locations and optimize resource allocation accordingly. Stores with higher sales should be prioritized for inventory restocks and marketing efforts, while those with lower sales can benefit from targeted promotions and additional support. By recognizing these patterns, the business can adjust its strategies to enhance both overall sales and individual store performance.

Through time series analysis, seasonal patterns by month and day of the week can be identified, enabling the development of tailored strategies to boost sales during peak periods. Additionally, cross-sectional analysis of top-selling products can reveal opportunities for creating effective product bundles and targeted promotions for specific categories, driving further sales growth.

6.5.1. Query

The query utilizes Views, CTEs, aggregation, and window functions for its development.

```
SELECT
s.Store_ID, s.Store_Name,
count(t.Transaction_ID) as Number_purchases,
sum(st.sales_per_transaction) as total_sales,
round(SUM(st.sales_per_transaction)/count(t.Transaction_ID),2) as average_per_transaction
FROM coffeeshowunfc.store s
LEFT JOIN coffeeshowunfc.transaction t on t.store_id = s.store_id
LEFT JOIN coffeeshowunfc.sales_transaction st on st.Transaction_ID = t.Transaction_ID
GROUP BY s.Store_ID, s.Store_Name
ORDER BY total_sales DESC;
```

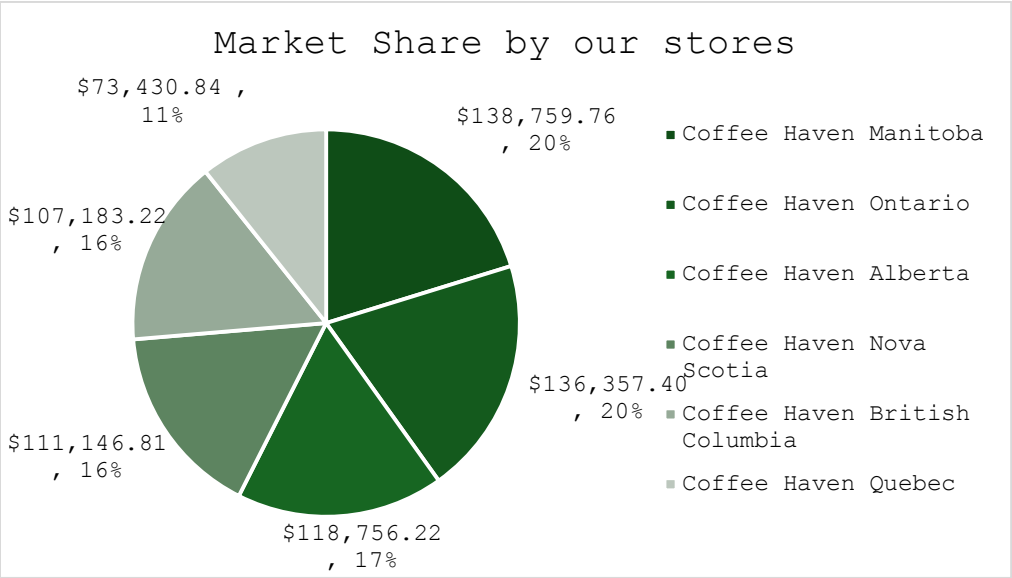
```
WITH sales_by_store as (
    SELECT
        tr.Transaction_Date as date,
        EXTRACT(YEAR FROM tr.Transaction_Date) as years,
        EXTRACT(MONTH FROM tr.Transaction_Date) as months,
        dayname(tr.Transaction_Date) as days,
        EXTRACT(HOUR FROM tr.Transaction_Date) as hours,
        tr.Transaction_ID,
        str.Store_Name as name_store,
        str.Store_ID as StoreID,
        tri.Quantity as quantity,
        tri.Price_per_Unit as prices
    FROM coffeeshowunfc.customer cus
    JOIN coffeeshowunfc.transaction tr ON cus.customer_id=tr.customer_id
    JOIN coffeeshowunfc.transaction_item tri ON tr.transaction_id=tri.transaction_id
    JOIN coffeeshowunfc.store str ON tr.Store_ID=str.Store_ID
)
```

```sql
SELECT
    StoreID,
    name_store,
    months,
    days,
    hours,
    sum(prices*quantity) as total_sales_by_store
FROM sales_by_store
GROUP BY StoreID,name_store,months,days, hours
ORDER BY StoreID, months, days, hours asc
```

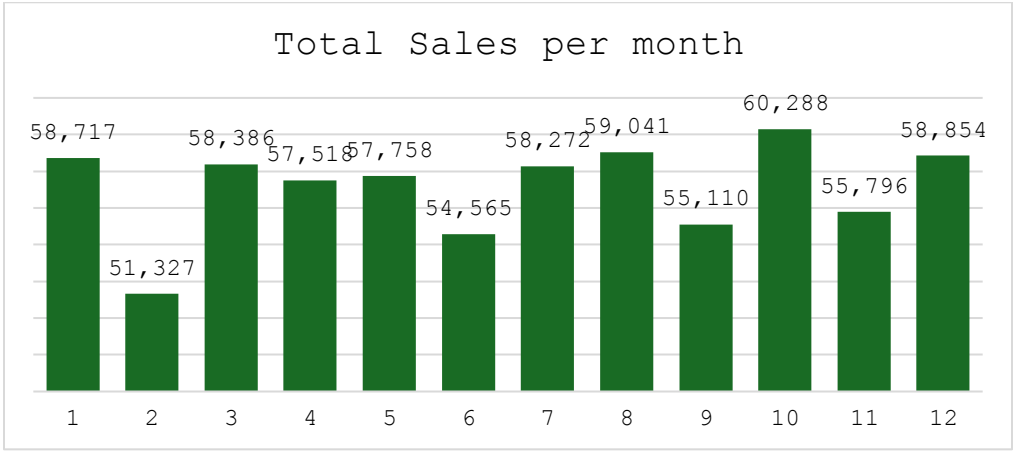
```sql
SELECT
    tr.Transaction_ID,
    EXTRACT(YEAR FROM tr.Transaction_Date) as years,
    EXTRACT(MONTH FROM tr.Transaction_Date) as months,
    DAYNAME(tr.Transaction_Date) as days,
    (Select Store_Name from store where Store_ID=tr.Store_ID) as Store_Name,
    tri.Product_ID,
    (Select Product_Name from product where Product_ID=tri.Product_ID) as Product_Name,
    tri.Quantity,
    tri.Price_per_Unit,
    (tri.Quantity*tri.Price_per_Unit) as total_by_product,
    (tri.Quantity*tri.Price_per_Unit) as avg_by_product,
    (Select Location_Province  from store a join location b on a.code=b.code where Store_ID=tr.Store_ID ) as Location_Province,
    (Select Location_City  from store a join location b on a.code=b.code where Store_ID=tr.Store_ID ) as Location_City,
    (Select Location_Postal_Code  from store a join location b on a.code=b.code where Store_ID=tr.Store_ID ) as Location_Postal_
FROM coffeeshowunfc.transaction tr
JOIN coffeeshowunfc.transaction_item tri ON tr.transaction_id=tri.transaction_id
JOIN coffeeshowunfc.store str ON tr.Store_ID=str.Store_ID
JOIN coffeeshowunfc.location loc ON str.code=loc.code
ORDER BY Location_Province, Store_Name, months, days;
```

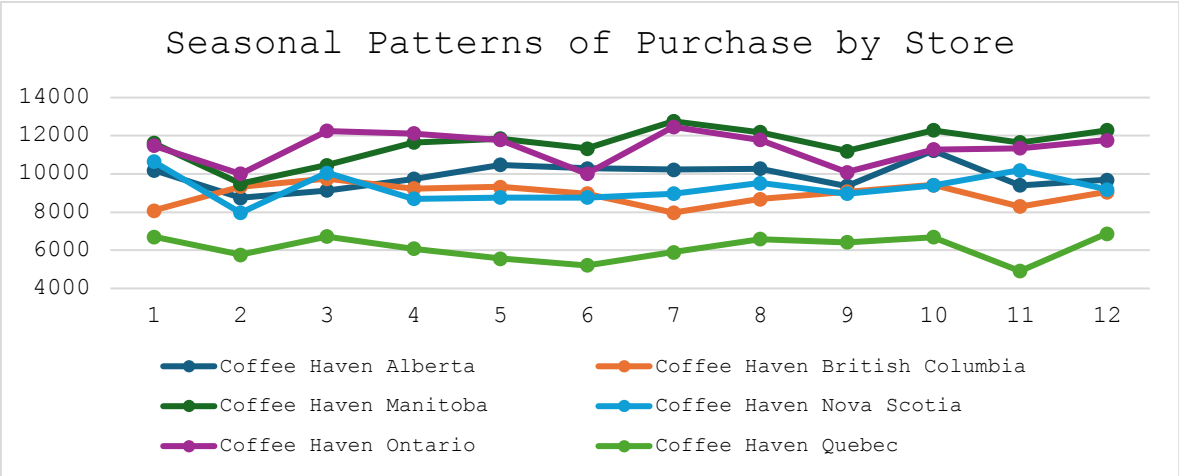
6.5.2. Results

Market share data reveals the performance of each store location, highlighting Manitoba and Ontario as the top performers, each contributing 20% of total sales. In contrast, Quebec requires further analysis to better understand consumer behavior, product mix, and customer engagement.



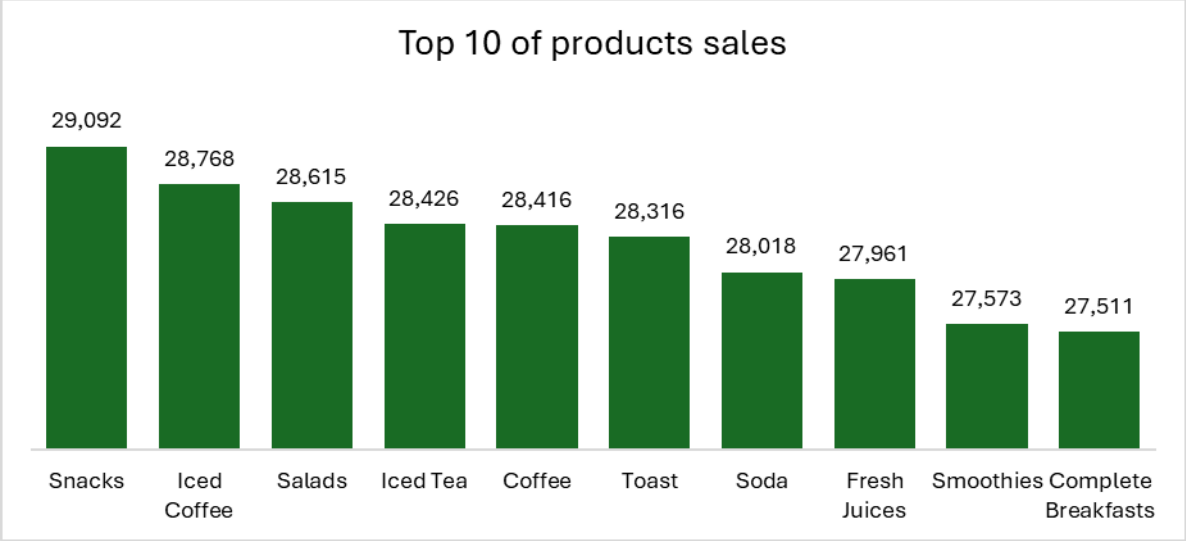
By analyzing historical data grouped by month, it becomes clear that February, June, and September require focused marketing strategies to maintain and boost sales during these periods. Additionally, evaluating and comparing the performance of each store throughout the year allows for more granular insights into specific trends, providing opportunities to refine strategies and improve overall results.





Understanding performance by category and product is essential for gaining deeper insights into consumer behavior. It allows businesses to identify key anchor categories that can drive growth across other product areas. By focusing on these high-performing categories, businesses can leverage their success to boost the sales of related or complementary products. A cross-analysis with daily sales data further enhances this strategy by helping to identify the best times to apply promotions. A heat map analysis revealed that Wednesdays consistently show lower sales compared to other days of the week, indicating a potential opportunity to boost sales on this slower day. This insight allows for the strategic bundling of key products and the timely application of promotions, ensuring they are effectively

targeted to maximize consumer interest and drive sales growth



Month	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
January	$ 4.331	$ 3.429	$ 4.199	$ 2.937	$ 3.873	$ 3.633	$ 2.266
February	$ 3.411	$ 2.406	$ 2.704	$ 3.694	$ 3.137	$ 2.951	$ 2.993
March	$ 3.055	$ 3.448	$ 3.023	$ 3.025	$ 4.136	$ 3.998	$ 3.796
April	$ 4.086	$ 3.485	$ 3.104	$ 3.146	$ 3.131	$ 3.642	$ 3.267
May	$ 3.482	$ 2.635	$ 3.170	$ 4.602	$ 4.188	$ 2.763	$ 3.033
June	$ 3.284	$ 3.490	$ 2.454	$ 2.115	$ 3.264	$ 3.706	$ 4.005
July	$ 3.289	$ 4.002	$ 3.449	$ 3.363	$ 3.582	$ 3.520	$ 3.198
August	$ 2.845	$ 3.480	$ 2.843	$ 4.036	$ 4.312	$ 3.960	$ 2.484
September	$ 4.412	$ 2.888	$ 2.800	$ 2.540	$ 3.119	$ 2.703	$ 3.830
October	$ 3.597	$ 3.685	$ 3.689	$ 4.152	$ 3.543	$ 3.057	$ 2.921
November	$ 3.031	$ 2.802	$ 2.548	$ 2.826	$ 4.025	$ 4.050	$ 3.408
December	$ 4.328	$ 3.396	$ 3.080	$ 3.674	$ 2.443	$ 2.885	$ 4.408

6.6. Reward Analysis

To encourage customer loyalty, one of the strategies implemented was the creation of a reward system for frequent buyers. Each time a customer purchases a product; they accumulate points that can later be redeemed for rewards. There are three types of rewards: Bonus Points, which offer benefits within the coffee shop such as early access to new menu

items or exclusive merchandise; Discounts, which can be applied to future purchases; and Free Items, which include a variety of products available at the coffee shop.

6.6.1. Query

The coffee shop wants to analyze the highest rewards that people usually redeem.

```
use coffeeshowunfc;
SELECT * FROM coffeeshowunfc.customer_reward;
SELECT
lr.Reward_ID,
count(cr.Reward_ID) as rewards_redeemed
FROM coffeeshowunfc.customer_reward cr
right join coffeeshowunfc.loyalty_reward  lr on cr.Reward_ID=lr.Reward_ID
group by lr.Reward_ID
order by rewards_redeemed desc;
```

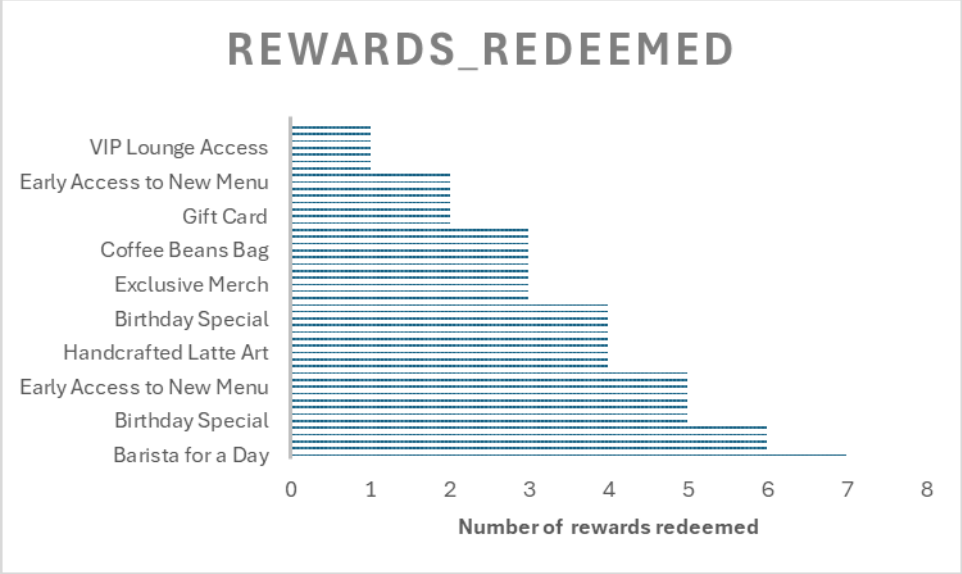
Additionally, identify the proportion of reward types and preferences chosen by a customer

```
with cte_reward as (
SELECT
cr.Customer_ID, lr.Reward_Type,
sum(cr.reward_quantity) over(PARTITION BY lr.reward_type) as total_by_type,
cr.reward_quantity,
(cr.reward_quantity/(sum(cr.reward_quantity) over(PARTITION BY cr.Customer_ID))) as percentage_per_customer,
rank () over (partition by cr.Customer_ID order by cr.reward_quantity DESC) as ranking
FROM coffeeshowunfc.customer_reward cr
join coffeeshowunfc.loyalty_reward lr on cr.Reward_ID = lr.Reward_ID
order by customer_id asc
)
select
customer_Id,reward_type, percentage_per_customer, ranking
from cte_reward
where ranking=1;
```

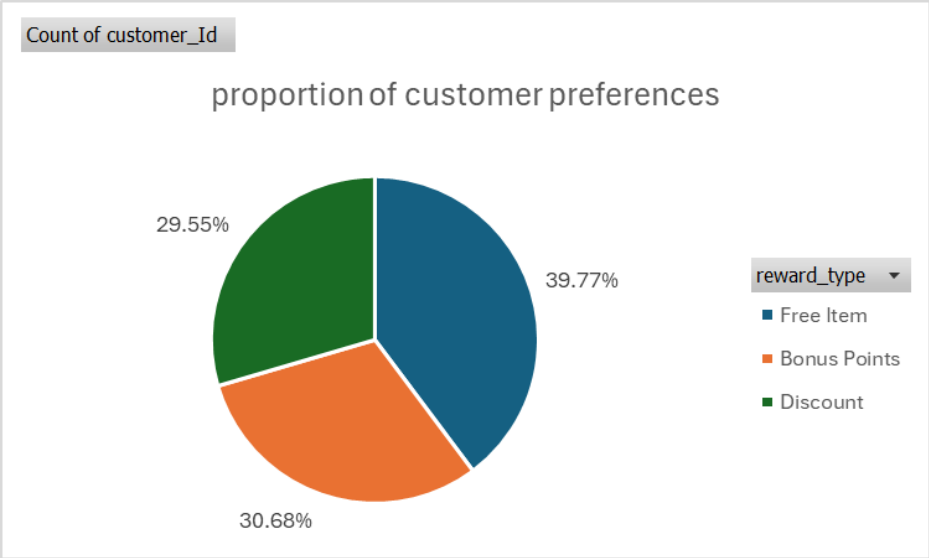
6.6.2. Results

"Barista for a Day" is the most frequently redeemed reward among all options offered in the loyalty program. This reward, classified under the "Free Item" category, grants customers the unique experience of stepping behind the counter and learning basic barista skills alongside the coffee shop staff. Its popularity suggests that customers are not only motivated by tangible benefits like discounts or products but also by experiential rewards

that create a deeper connection with the brand.



Of the total number of customers who redeemed rewards using points, 39.77% of them tend to choose first the “Free points” type over the others, 29.55% the “Discount” type, and 30.68% the “Bonus Points” type.



7. Data Control Language (DCL)

Used by the database administrator for management and security of the database. Two users were created, and privileges were assigned.

For the reporting user privilege functions like Select to the table transactions, and transaction items were granted.

```
use coffeeshowunfc;
SHOW GRANTS;
-- create user - Password:Test123
create user 'reporting_user'@'localhost' IDENTIFIED BY 'Test123';
SHOW GRANTS for 'reporting_user'@'localhost';
GRANT SELECT ON coffeeshowunfc.transaction TO 'reporting_user'@'localhost';
GRANT SELECT ON coffeeshowunfc.transaction_item TO 'reporting_user'@'localhost';
```

For the user manager the administrator granted privilege functions like Select, insert and update to the employee table, understanding security and seniority level, this user is allowed to access this information.

```
/*manager of store right only SELECT,INSERT,UPDATE*/
use coffeeshowunfc;
create user 'manager'@'localhost' IDENTIFIED BY 'Test123';
GRANT SELECT,INSERT,UPDATE ON coffeeshowunfc.employee TO 'manager'@'localhost';
SHOW GRANTS for 'manager'@'localhost';
```

The administrator revoked all privileges to the reporting user as he left the company.

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'reporting_user'@'localhost';
```

8. Conclusions

This project allowed us to apply theoretical knowledge to a real-world business scenario by designing, implementing, and analyzing a relational database for a coffee shop. Through understanding the company, creating the model, inserting the information, and analyzing the data, were developed a comprehensive understanding of how database systems support data integrity, reduce redundancy, and facilitate meaningful insights.

The implementation of DDL, DML, DQL, and DCL statements shows the ability to manage a database environment in a structured and efficient manner. Furthermore, the use of advanced queries enabled the identification of customer behavior patterns, store performance metrics, and opportunities for business growth.

Overall, this project reinforced the importance of database design in modern data-driven environments and highlighted the role of SQL as a foundational tool for both data management and strategic decision-making in the context of business analytics.

9. References

Chiu, C.-M., Hsu, M.-H., Lai, H., & Chang, C.-M. (2012). Re-examining the influence of trust on online repeat purchase intention: The moderating role of habit and its antecedents. Decision Support Systems, 53(4), 835–845. doi:10.1016/j.dss.2012.05.021