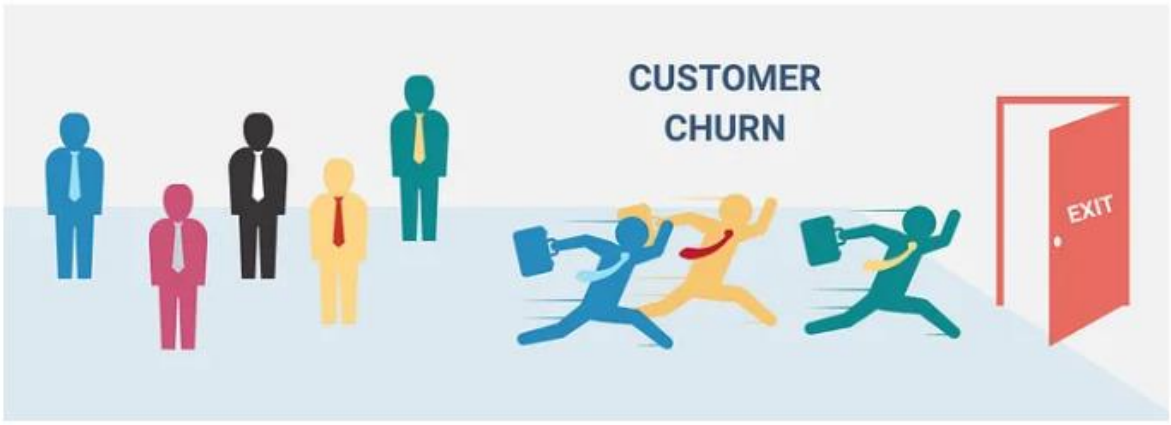

Project Title

E-Commerce Customer Churn Analysis

V.Hemapriya

Contents Page No

Overview of MySQL Workbench

MySQL Workbench is a visual database design and administration tool for MySQL databases. It allows you to create, modify, and manage database schemas and design and execute SQL queries. MySQL Workbench also provides tools for performance tuning and server administration.

Introduction

In understanding customer churn has become essential for businesses striving to enhance customer retention and satisfaction. Customer churn refers to the rate at which customers cease their relationship with a company or stop using their service, often switching to a competitor or discontinuing their purchases altogether. Analysis of customer churn can reveal critical insights that enable businesses to create targeted strategies, ultimately reducing churn and fostering long-term customer loyalty.

Problem Statement:

In the realm of e-commerce, businesses face the challenge of understanding customer churn patterns to ensure customer satisfaction and sustained profitability. This project aims to delve into the dynamics of customer churn within an e-commerce domain, utilizing historical transactional data to uncover underlying patterns and drivers of churn. By analyzing customer attributes such as tenure, preferred payment modes, satisfaction scores, and purchase behaviour, the project seeks to investigate and understand the dynamics of customer attrition and their propensity to churn. The ultimate objective is to equip e-commerce enterprises with actionable insights to implement targeted retention strategies and mitigate churn, thereby fostering long-term customer relationships and ensuring business viability in a competitive landscape.

1.Project Steps and Objectives:

1.1Data Cleaning:

1.1.1 Handling Missing Values and Outliers:

Impute mean for the following columns and round off to the nearest integer if required:

- ✓ WarehouseToHome, HourSpendOnApp, OrderAmountHikeFromlastYear, DaySinceLastOrder.
- ✓ Tenure, Coupon Used, Order Count.

```
-- WarehouseToHome
set sql_safe_updates = 0;
set @avg_WarehouseToHome=(select round(avg(WarehouseToHome))as avg_WarehouseToHome
from customer_churn);

update customer_churn
set WarehouseToHome = @avg_WarehouseToHome
where WarehouseToHome is null;

-- HourSpendOnApp
set @avg_HourSpendOnApp=(select round(avg(HourSpendOnApp))as avg_HourSpendOnApp
from customer_churn);

update customer_churn
set HourSpendOnApp = @avg_HourSpendOnApp
where HourSpendOnApp is null;


-- OrderAmountHikeFromlastYear
set @avg_OrderAmountHikeFromlastYear=(select round(avg(OrderAmountHikeFromlastYear))as
OrderAmountHikeFromlastYear from customer_churn);

update customer_churn
set OrderAmountHikeFromlastYear = @avg_OrderAmountHikeFromlastYear
where OrderAmountHikeFromlastYear is null;

-- DaySinceLastOrder
set @avg_DaySinceLastOrder=(select round(avg(DaySinceLastOrder))as avg_DaySinceLastOrder
from customer_churn);

update customer_churn
set DaySinceLastOrder = @avg_DaySinceLastOrder
where DaySinceLastOrder is null;
```

```sql
-- Tenure
set @avg_Tenure=(select round(avg(Tenure))as avg_Tenure
from customer_churn);

update customer_churn
set Tenure = @avg_Tenure
where Tenure is null;

-- CouponUsed
set @avg_CouponUsed=(select round(avg(CouponUsed))as avg_CouponUsed
from customer_churn);

update customer_churn
set CouponUsed = @avg_CouponUsed
where CouponUsed is null;

-- OrderCount
set @avg_OrderCount=(select round(avg(OrderCount))as avg_OrderCount
from customer_churn);

update customer_churn
set OrderCount = @avg_OrderCount
where OrderCount is null;

-- WarehouseToHome deleting rows where > 100
delete from customer_churn
where WarehouseToHome > 100;
```

1.1.2 Dealing with Inconsistencies:

- ✓ Replace occurrences of “Phone” in the 'PreferredLoginDevice' column and “Mobile” in the 'PreferedOrderCat' column with “Mobile Phone” to ensure uniformity.
- ✓ Standardize payment mode values: Replace "COD" with "Cash on Delivery" and "CC" with "Credit Card" in the PreferredPaymentMode column.

```
-- b)dealing with inconsistencies
update customer_churn
set PreferredLoginDevice = if(PreferredLoginDevice ='Phone','Mobile Phone',PreferredLoginDevice);

update customer_churn
set PreferedOrderCat= if(PreferedOrderCat ='Mobile','Mobile Phone',PreferedOrderCat);

update customer_churn
set PreferredPaymentMode = if(PreferredPaymentMode = 'CC','Credit Card',PreferredPaymentMode);

update customer_churn
set PreferredPaymentMode = if(PreferredPaymentMode = 'COD','Cash On Delivery',
PreferredPaymentMode);
```

2.Data Transformation:

2.1 Column Renaming:

```
-- a)Column Renaming:

alter table customer_churn
rename column PreferedOrderCat to PreferredOrderCat;
alter table customer_churn
rename column HourSpendOnApp to HoursSpentOnApp;
```

2.2 Creating New Columns:

```
-- b)Creating New Columns:

alter table customer_churn
add column ComplaintReceived enum ('Yes','No'),
add column ChurnStatus enum ('Churned','Active');

-- set the values for the new column
set sql_safe_updates = 0;
update customer_churn
set ComplaintReceived = if(Complain = 1, 'Yes','No'),
    ChurnStatus = if(Churn = 1,'Churned','Active');
```

2.3 Column Dropping:

```
-- c)column dropping
alter table customer_churn
drop column Churn;
alter table customer_churn
drop column Complain;
```

3.Data Exploration and Analysis

Query 1:

```
/* 1. Retrieve the count of churned and active customers from the dataset.*/
```

```
select ChurnStatus,count(*) as customer_count
from customer_churn
group by ChurnStatus;
```

Result:

	ChurnStatus	customer_count
▶	Active	4680
	Churned	948

Query 2:

```
/* 2.Display the average tenure and total cashback amount of customers who churned.*/
```

```
select ChurnStatus,avg(Tenure) as avg_tenure,
sum(CashbackAmount) as total_cashbackamount
from customer_churn
where ChurnStatus = 'churned'
group by ChurnStatus;
```

Result:

	ChurnStatus	avg_tenure	total_cashbackamount
▶	Churned	3.9451	152030

Query 3:

```
/*3. Determine the percentage of churned customers who complained.*/
```

```
select ComplaintReceived,
concat(round(count(*)/(select count(*) from customer_churn)*100,2),'%')
as Churned_percentage
from customer_churn
group by ComplaintReceived;
```

Result:

ComplaintReceived	Churned_percentage
Yes	28.50%
No	71.50%

Query 4:

```
/*4.  Find the gender distribution of customers who complained.*/
```

```
select Gender,count(*) as complaint_count
from customer_churn
where ComplaintReceived = 'Yes'
group by Gender;
```

Result:

Gender	complaint_count
Female	690
Male	914

Query 5:

```
/* 5. Identify the city tier with the highest number of churned customers whose preferred
order category is Laptop & Accessory*/

select CityTier,count(*) from customer_churn
where ChurnStatus = 'Churned' and PreferredOrderCat='Laptop & Accessory'
group by CityTier
order by CityTier desc limit 1;
```

Result:

	CityTier	count(*)
▶	3	150

Query 6:

```
/*6. Identify the most preferred payment mode among active customers.*/

select PreferredPaymentMode,count(*) as active_customers from customer_churn
where ChurnStatus = 'Active'
group by PreferredPaymentMode
order by active_customers desc limit 1;
```

Result:

	PreferredPaymentMode	active_customers
▶	Debit Card	1956

Query 7:

```
/* 7.  Calculate the total order amount hike from last year for customers who are single
 and prefer mobile phones for ordering.*/

select sum(OrderAmountHikeFromlastYear)
as total_order_amount_hike
from customer_churn
where PreferredOrderCat = 'Mobile Phone'
and MaritalStatus ='Single';
```

Result:

	total_order_amount_hike
▶	12177

Query 8:

```
/* 8.  Find the average number of devices registered among customers who used UPI as
their preferred payment mode.*/

select NumberOfDeviceRegistered, avg(NumberOfDeviceRegistered) as avg_NoOfdevice
from customer_churn
where PreferredPaymentMode = 'UPI'
group by NumberOfDeviceRegistered
order by PreferredPaymentMode ;
```

Result:

NumberOfDeviceRegistered	avg_NoOfdevice
4	4.0000
3	3.0000
2	2.0000
5	5.0000
1	1.0000
6	6.0000

Query 9:

```
/*9.  Determine the city tier with the highest number of customers.*/

select CityTier,count(*) as total_count from customer_churn
group by CityTier
order by CityTier limit 1;
```

Result:

CityTier	total_count
1	3666

Query 10:

```
/*10.  Identify the gender that utilized the highest number of coupons.*/
```

```
select Gender,count(CouponUsed) as count_CouponUsed from customer_churn
group by Gender
order by Gender limit 1;
```

Result:

	Gender	count_CouponUsed
▶	Male	3382

Query 11:

```
/*11. List the number of customers and the maximum hours spent on the app in each
preferred order category*/
```

```
select PreferredOrderCat,count(CustomerID) as customer_count ,
max(HoursSpentOnApp) as Maximum_hours
from customer_churn
group by PreferredOrderCat
order by Maximum_hours;
```

Result:

	PreferredOrderCat	customer_count	Maximum_hours
▶	Others	264	4
	Grocery	410	4
	Laptop & Accessory	2050	5
	Mobile Phone	2078	5
	Fashion	826	5

Query 12:

```
/* 12.Calculate the total order count for customers who prefer using credit cards and
have the maximum satisfaction score.*/

select sum(OrderCount) as total_ordercount ,
max(SatisfactionScore) as Maximum_Satisfication_Score from customer_churn
where PreferredPaymentMode = 'Credit Card';
```

Result:

	total_ordercount	Maximum_Satisfication_Score
▶	5505	5

Query 13:

```
/* 13. How many customers are there who spent only one hour on the app and days
since their last order was more than 5?
*/
select HoursSpentOnApp,count(*) as total_count from customer_churn
where HoursSpentOnApp <= 1 and DaySinceLastOrder > 5
group by HoursSpentOnApp
order by HoursSpentOnApp;
```

Result:

	HoursSpentOnApp	total_count
▶	1	8

Query 14:

```
/* 14.What is the average satisfaction score of customers who have complained?
*/
select ComplaintReceived,round(avg(SatisfactionScore)) as Avg_SatisfactionScore
from customer_churn
where ComplaintReceived = 'Yes'
group by ComplaintReceived
order by ComplaintReceived ;
```

Result:

	ComplaintReceived	Avg_SatisfactionScore
▶	Yes	3

Query 15:

```
/*15.List the preferred order category among customers who used more than 5
coupons*/
select PreferredOrderCat,count(*) as total_count from customer_churn
where CouponUsed > 5
group by PreferredOrderCat
order by total_count;
```

Result:

PreferredOrderCat	total_count
Others	28
Grocery	42
Mobile Phone	45
Fashion	89
Laptop & Accessory	99

Query 16:

```
/*16.  List the top 3 preferred order categories with the highest average cashback
amount.*/
select PreferredOrderCat,round(avg(CashbackAmount)) as Avg_CashbackAmount
from customer_churn
group by PreferredOrderCat
order by  Avg_CashbackAmount desc limit 3;
```

Result:

PreferredOrderCat	Avg_CashbackAmount
Others	304
Grocery	266
Fashion	210

Query 17:

```
/*17.Find the preferred payment modes of customers whose average tenure is 10
months and have placed more than 500 orders.*/

select PreferredPaymentMode,avg(Tenure) as Avg_Tenure,count(OrderCount)
from customer_churn
group by PreferredPaymentMode
order by Avg_Tenure desc limit 3;
```

Result:

PreferredPaymentMode	Avg_Tenure	count(OrderCount)
E wallet	10.3860	614
Debit Card	10.3607	2312
Credit Card	10.2249	1774

Query 18:

```
/* 18.Categorize customers based on their distance from the warehouse to home such
as 'Very Close Distance' for distances <=5km, 'Close Distance' for <=10km,
'Moderate Distance' for <=15km, and 'Far Distance' for >15km. Then, display the
churn status breakdown for each distance category.
*/
select
case
  when  WarehouseToHome <= 5 then 'Very Close Distance'
  when WarehouseToHome  <= 10 then 'Close Distance'
  when  WarehouseToHome <= 15 then 'Moderate Distance'
  else 'Far Distance'
  end as Category_distance,ChurnStatus,count(*) as customer_count
  from customer_churn
  group by Category_distance,ChurnStatus
  order by Category_distance,ChurnStatus;
```

Result:

Category_distance	ChurnStatus	customer_count
Close Distance	Churned	265
Close Distance	Active	1696
Far Distance	Churned	498
Far Distance	Active	1871
Moderate Distance	Churned	184
Moderate Distance	Active	1106
Very Close Distance	Churned	1
Very Close Distance	Active	7

Query 19:

```
/* List the customer's order details who are married, live in City Tier-1, and their
order counts are more than the average number of orders placed by all
customers
*/
-- without using CTE
select CustomerID ,round(avg(OrderCount)) as Avg_ordercount from customer_churn
where MaritalStatus = 'Married' and CityTier = 1
group by CustomerID
having Avg_ordercount > (select avg(OrderCount) from customer_churn)
order by Avg_ordercount ;

-- using CTE
with average_order_count as (select ROUND(avg(OrderCount))
 as Avg_order from customer_churn)

select CustomerID,round(avg(OrderCount)) as Order_count from customer_churn
where MaritalStatus = 'Married' and Citytier = 1
group by CustomerID
having (select Avg_order from average_order_count) < Order_count
order by Order_count ;
```

Result for with and without using CTE is same:

CustomerID	Avg_ordercount
50119	4
50367	4
50385	4
50453	4
50727	4
50734	4
51060	4
51084	4
51198	4
51227	4

Query 20:

```
/*20 a) Create a 'customer_returns' table in the 'ecomm' database and insert the
following data
*/
create table customer_returns(
ReturnID int,
CustomerID int,
ReturnDate date,
RefundedAmount int
);

insert into customer_returns(ReturnID,CustomerID,ReturnDate,RefundedAmount)
values
(1001, 50022, '2023-01-01', 2130),
(1002, 50316, '2023-01-23', 2000),
(1003, 51099, '2023-02-14', 2290),
(1004, 52321, '2023-03-08', 2510),
(1005, 52928, '2023-03-20', 3000),
(1006, 53749, '2023-04-17', 1740),
(1007, 54206, '2023-04-21', 3250),
(1008, 54838, '2023-04-30', 1990);
```

Result:

ReturnID	CustomerID	ReturnDate	RefundedAmount
1001	50022	2023-01-01	2130
1002	50316	2023-01-23	2000
1003	51099	2023-02-14	2290
1004	52321	2023-03-08	2510
1005	52928	2023-03-20	3000
1006	53749	2023-04-17	1740
1007	54206	2023-04-21	3250
1008	54838	2023-04-30	1990
NULL	NULL	NULL	NULL

Query 21:

```
/*b)Display the return details along with the customer details of those who have
churned and have made complaints
*/
select
c.CustomerID,
c.ComplaintReceived,
c.ChurnStatus,
r.ReturnID,
r.ReturnDate,
r.RefundedAmount
from customer_churn c
join customer_returns r on c.CustomerID = r.CustomerID
where ComplaintReceived = 'Yes' and ChurnStatus = 'Churned';
```

Result:

CustomerID	ComplaintReceived	ChurnStatus	ReturnID	ReturnDate	RefundedAmount
50316	Yes	Churned	1002	2023-01-23	2000
52321	Yes	Churned	1004	2023-03-08	2510
53749	Yes	Churned	1006	2023-04-17	1740

4.Conclusion:

Insights

- The dataset includes 5,628 customers and the overall churn rate is 16.78%, indicating a significant portion of customers disengaging with the platform.
- Cash on delivery has the highest churn rate, suggesting a need to enhance the experience for customers using this method. Also, E-wallet and UPI users also show relatively high churn rates, signalling potential issues with the payment process or overall satisfaction.
- Customers using computers for log-in show a higher churn rate compared to those using phones.
- Customers with a tenure of 0–6 months show a higher churn rate, indicating a challenge in retaining customers during the initial phases. As the tenure increases, the churn rate consistently decreases, reaching 0.0% for customers with more than 2 years association.
- Tier 3 cities have the highest churn rate, indicating potential challenges or dissatisfaction specific to these regions. Also, Tier 1 and Tier 2 cities also experience notable churn rates, requiring targeted strategies for each tier.
- Customers with 6 registered devices have the highest churn rate, possibly due to complex user experiences or account management challenges, while users with fewer registered devices show relatively lower churn rates.

- Customers with moderate and far distances have higher churn rates, indicating potential issues with delivery services.
- Churn rates increase with lower satisfactory scores, emphasizing the importance of customer satisfaction.
- Mobile and fashion categories show higher churn rates, signalling potential issues or competition in these segments. Focusing on enhancing the user experience in these categories may help retain customers.

Recommendation

1. Improve the Cash on Delivery experience by streamlining the process. Investigate issues with E-wallet and UPI payments to enhance overall satisfaction and trust in the payment system.

2. Optimize the user experience for computer users, ensuring a seamless and user-friendly shopping experience. Also, consider offering exclusive promotions or features for computer users to boost engagement.

3. Focus on strategies and for customers in the 0–6 months tenure range. Implement loyalty programs and exclusive perks to encourage long-term association. Acknowledge and reward customer loyalty for those exceeding 2 years.

4. Develop targeted strategies for Tier 3 cities to address specific challenges or concerns. Tailor marketing and service offerings based on the unique characteristics of each tier. Regularly assess and adapt strategies to reduce churn in Tier 1 and Tier 2 cities.

5. Simplify user experiences for customers with 6 registered devices. Provide clear instructions for account management and consider streamlining features.

6. Optimize delivery services in moderate and far-distance areas. Address potential issues causing higher churn rates, such as delivery times, tracking accuracy, or customer communication. Consider partnerships or improvements to expedite deliveries.

7. Invest in improving user experiences in the mobile and fashion categories. Identify and resolve issues related to product quality and pricing. Implement marketing strategies to highlight unique selling points and capture customer interest in these categories.