CREDIT CARD

Weekly Status Report

CONTENT

- Project Objective
- Basic Requirements
- Tables Details
- SQL and Python Connection
- SQL and Power BI Connection
- SQL Queries and DAX
- Dashboard and Insights

OBJECTIVE

To develop a comprehensive credit card weekly dashboard that provides real-time insights into key performance metrics and trends, enabling stakeholders to monitor and analyze credit card operations effectively.

BASIC REQUIREMENTS

1. Download Dataset
2. Create tables in SQL
3. import CSV file into SQL

Download The Dataset :

Table Details

Table 1: cc_detail

--> Table Details

```
[51]: # table 1: cc_detail
query = """
        SELECT * from cc_detail
    """
df =pd.DataFrame(execute_query(query), columns =cc_detail_cols)
df.shape
```

```
[51]: (10108, 21)
```

```
[9]: # first 5 Rows of cc_detail
query = """
        SELECT * from cc_detail
        limit 5
    """
df =pd.DataFrame(execute_query(query), columns =cc_detail_cols)
df
```

[9]:

	client_num	card_category	annual_fees	activation_30_days	customer_acq_cost	week_start_date	week_num	qtr	current_year	credit_limit	...	total_trans_amt
0	773155533	Blue	140	1	66	2023-09-24	Week-39	Q3	2023	3261.00	...	4739
1	717261708	Blue	125	1	54	2023-06-18	Week-25	Q2	2023	13457.00	...	1548
2	714894708	Blue	440	0	109	2023-05-07	Week-19	Q2	2023	23958.00	...	2777
3	820599333	Blue	125	0	106	2023-12-17	Week-51	Q4	2023	1438.30	...	4469
4	774588633	Blue	100	0	72	2023-10-01	Week-40	Q4	2023	1438.30	...	4928

5 rows × 21 columns

Table 1: cust_detail

```
[52]: # table 1: cust_detail
query = """
        SELECT * from cust_detail
    """
df =pd.DataFrame(execute_query(query), columns =cust_detail_cols)
df.shape
```

```
[52]: (10108, 17)
```

```
[53]: # first 5 Rows of cust_detail
query = """
        SELECT * from cust_detail
        limit 5
    """
df =pd.DataFrame(execute_query(query), columns =cust_detail_cols)
df
```

[53]:

	client_num	customer_age	gender	dependent_count	education_level	marital_status	state_cd	zipcode	car_owner	house_owner	personal_loan	contact	cu
0	713126733	51	F	1	High School	Married	CA	91750	no	no	no	cellular	Self
1	769576608	51	F	3	Graduate	Single	TX	91750	yes	yes	no	unknown	
2	709110633	48	F	4	Unknown	Married	FL	91750	yes	no	no	cellular	
3	754851783	43	M	4	Unknown	Single	IL	91750	no	no	yes	cellular	V
4	720409608	39	F	3	Graduate	Married	FL	91750	yes	yes	no	unknown	Self

SQL AND PYTHON CONNECTION

Import Lib..

[1]:
```
import pandas as pd
import numpy as np
import psycopg2
```

[2]:
```
# Database Configuration---
config= {'database': 'ccdb', 'user': 'postgres', 'password': 'kshma'}
```

[3]:
```
# Establish Connection
try:
     with psycopg2.connect(**config) as con:
            print('Connected to the PostgreSQL Server.')
except (psycopg2.DatabaseError, Exception) as error:
    print(error)
```

```
Connected to the PostgreSQL Server.
```

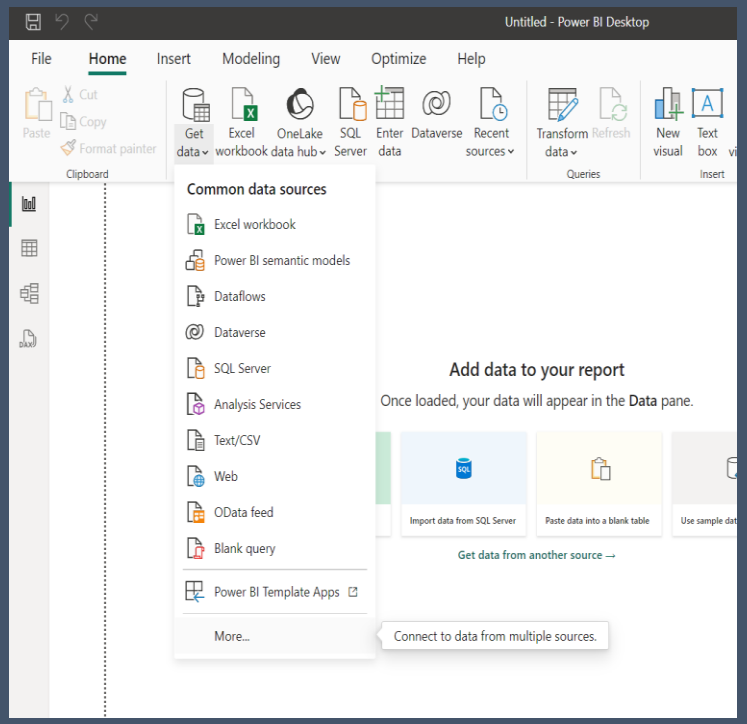
Success

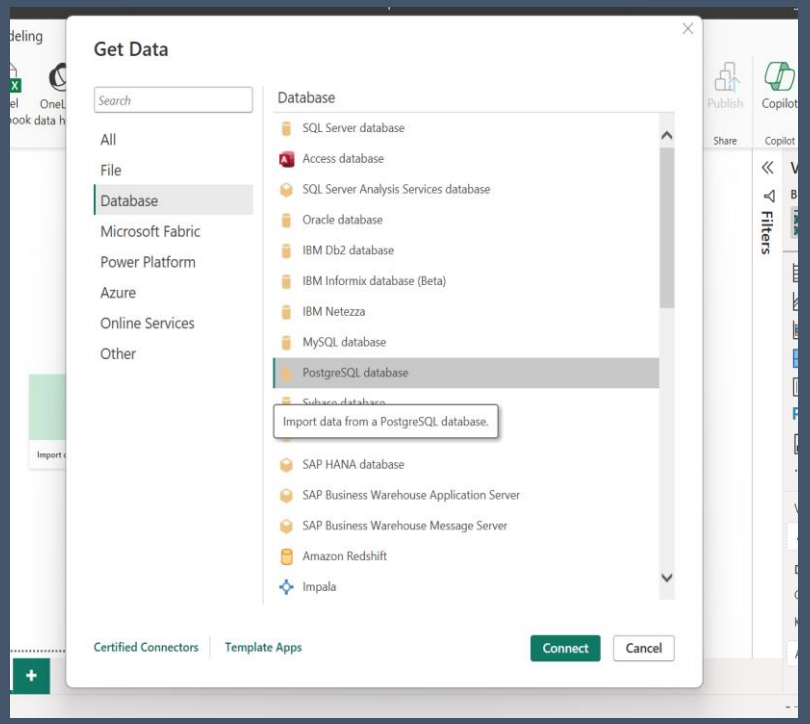
This function will help to execute all SQL queries

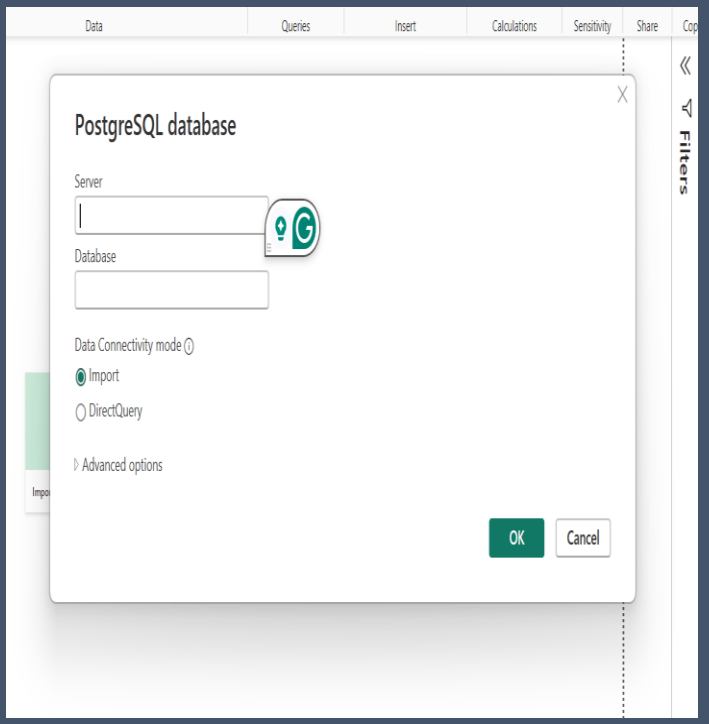
Function To Run the SQL Queries

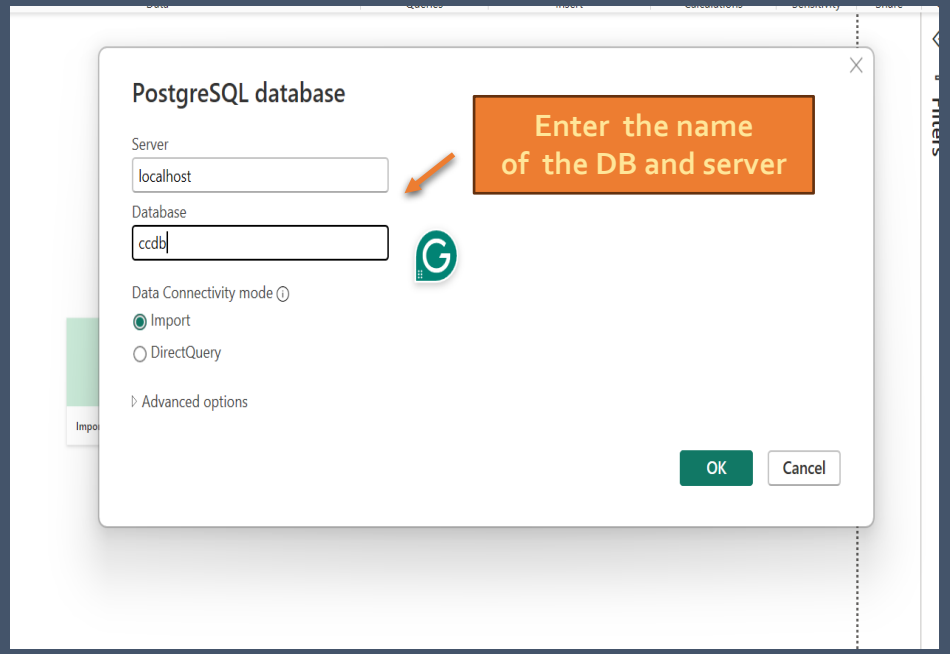
[8]:
```
# Creating cursor obj, execute, then Fetch Data
def execute_query(q1, commit=False):
    try:
        with con.cursor() as cur:
            cur.execute("BEGIN")
            cur.execute(q1)
            if commit:
                con.commit()
            else:
                data =cur.fetchall()
                return data
    except (psycopg2.DatabaseError, Exception) as error:
        print(error)
```

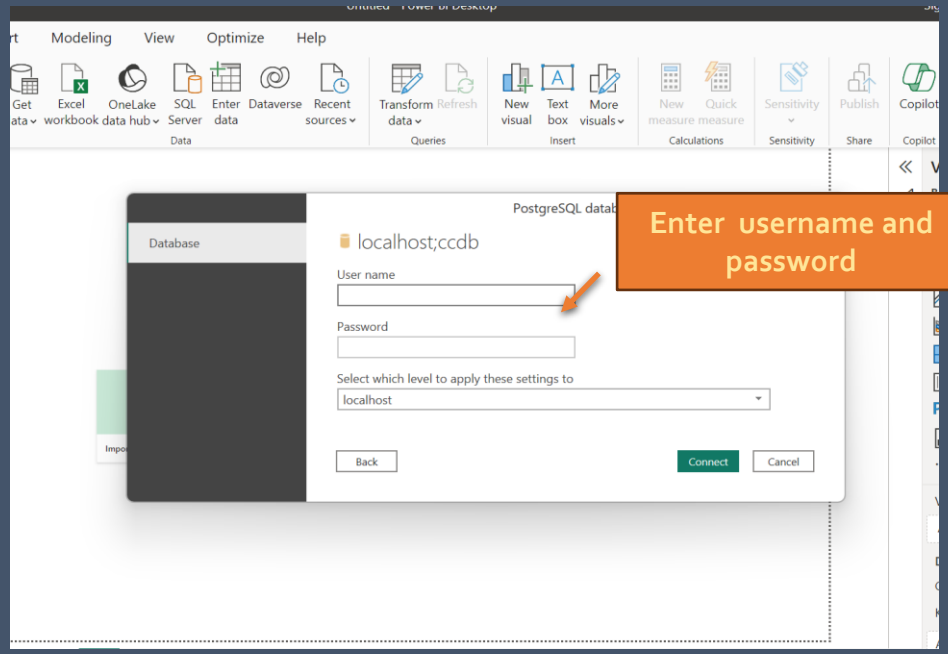
SQL AND POWER BI CONNECTION

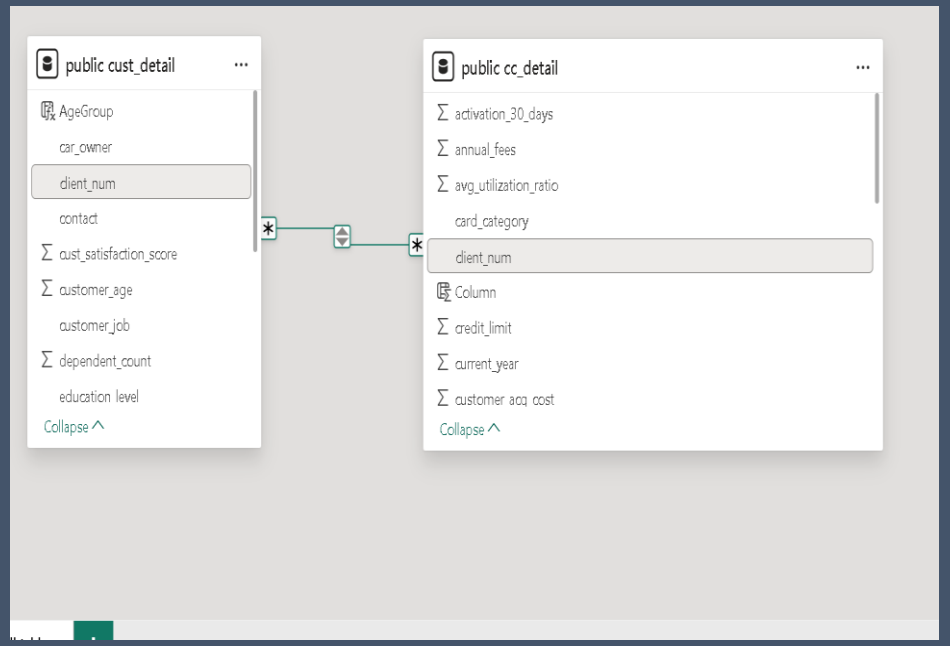


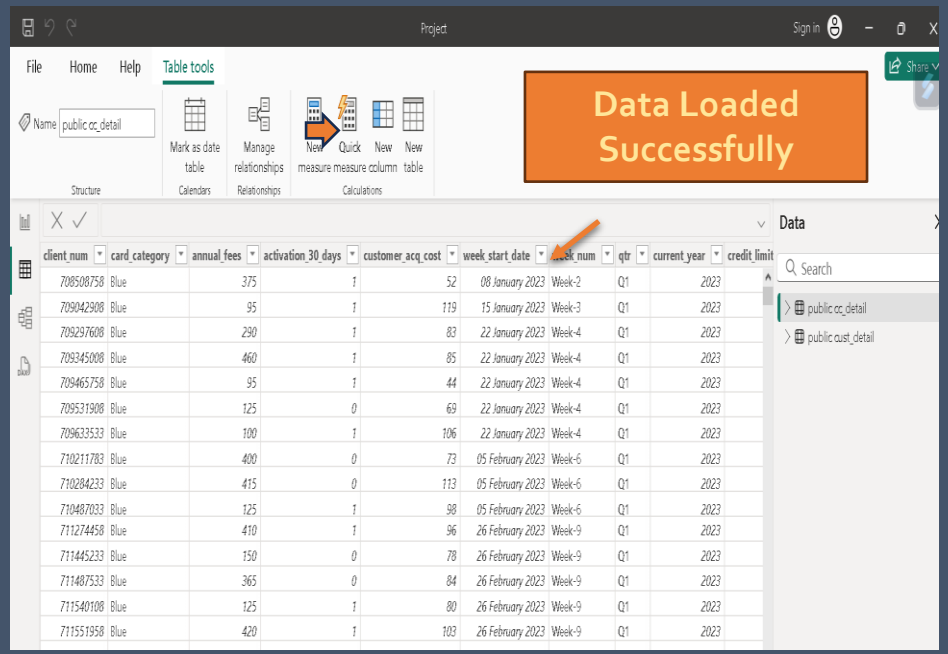





PostgreSQL database
Server
localhost
Database
ccdb
Enter the name of the DB and server
Data Connectivity mode
Import
DirectQuery
Advanced options
OK
Cancel


Modeling
View
Optimize
Help
Database
localhost;ccdb
User name
Password
Select which level to apply these settings to
localhost
Back
Connect
Cancel
Enter username and password


public cust_detail
AgeGroup
car_owner
client_num
contact
cust_satisfaction_score
customer_age
customer_job
dependent_count
education level
Collapse
public cc_detail
activation_30_days
annual_fees
avg_utilization_ratio
card_category
client_num
Column
credit_limit
current_year
customer_acq_cost
Collapse


Project
File
Home
Help
Table tools
Data Loaded Successfully
Data
Search
public cc_detail
public cust_detail

SQL QUERIES AND DAX

New columns added in cc_detail table

Adding New Columns

[21]:
```
# Creating and Adding data to Revenue in cc_details table
query ="""
    alter table cc_detail add column Revenue Numeric;
    update cc_detail set Revenue = Round((annual_fees+
    total_trans_amt + interest_earned), 0)
    """
execute_query(query, commit=True)
```

[19]:
```
# Creating and Adding data to WeekNum2 in cc_details table
query ="""
    alter table cc_detail add column WeekNum2 Numeric;
    update cc_detail set
    WeekNum2 = FLOOR((EXTRACT(DOY FROM week_start_date) -
            EXTRACT(DOY FROM DATE '2024-01-01')) / 7) + 1
    """
execute_query(query, commit=True)
```

[22]:
```
# Creating and Adding data to Profit in cc_details table
query ="""
    alter table cc_detail add column Profit Numeric;
    update cc_detail set
    Profit = Round(revenue-(customer_acq_cost + total_trans_amt),0 )
    """
execute_query(query, commit=True)
```

Formatting

```
1 revenue = 'public cc_detail'[annual_fees] +
2           'public cc_detail'[total_trans_amt] +
3           'public cc_detail'[interest_earned]
```

```
1 Profit = 'public cc_detail'[revenue]-
2          ('public cc_detail'[customer_acq_cost] +
3           'public cc_detail'[total_trans_amt])
```

```
1 WeekNum2 = WEEKNUM('public cc_detail'[week_start_date].[Date])
```

Adding New Columns - cust_detail

[112]:
```
# Creating and Adding data to AgeGroup  in cust_details table
query ="""
    alter table cust_detail add column AgeGroup Varchar;
    update cust_detail set
    AgeGroup = case
        when customer_age<30 then '20-30'
        when customer_age < 40 then '30-40'
        when customer_age < 50 then '40-50'
        when customer_age < 60 then '50-60'
        when customer_age >= 60 then '60+'
        else 'Unknown' end
    """
execute_query(query, commit=True)
```

[113]:
```
# Creating and Adding data to IncomeGroup in cust_details table
query ="""
    alter table cust_detail add column IncomeGroup Varchar;
    update cust_detail set
    IncomeGroup = case
        when income<10000 then 'Very Low'
        when income < 35000 then 'Low'
        when income < 70000 then 'Med'
        when income >=70000 then 'High'
        else 'Unknown' end
    """
execute_query(query, commit=True)
```

New Columns Added in cust_detail table

```
AgeGroup = SWITCH(
    TRUE(),
    'public cust_detail'[customer_age]< 30, "20-30",
    'public cust_detail'[customer_age] >=30 && 'public cust_detail'[customer_age] < 40, "30-40",
    'public cust_detail'[customer_age] >=40 && 'public cust_detail'[customer_age] < 50, "40-50",
    'public cust_detail'[customer_age] >=50 && 'public cust_detail'[customer_age] < 60, "50-60",
    'public cust_detail'[customer_age] >=60, "60+",
    "Unknown")
```

```
IncomeGroup = SWITCH(
    TRUE(),
    'public cust_detail'[income] < 10000, "Very Low",
    'public cust_detail'[income] >=10000 && 'public cust_detail'[income] < 35000, "Low",
    'public cust_detail'[income] >=35000 && 'public cust_detail'[income] < 70000, "Med",
    'public cust_detail'[income] >=70000, "High",
    "Unknown"
)
```

Key Performance Indicator

--> KPIs Calculation in cc_detail

[5]:
```
# Total Revenue, Trans Count, Trans. Amount, Interest,
# annual_fees, customer_acq_cost and Profit
query= """
    Select sum(revenue) as Revenue,
    sum(total_trans_ct) as trans_ct,
    sum(total_trans_amt) as trans_amt,
    sum(interest_earned) as Interest,
    sum(annual_fees) as annual_fees,
    sum(customer_acq_cost) as customer_acq_cost,
    sum(profit) as profit
    from cc_detail;
    """
output =execute_query(query)
print(f'Total Revenue: {output[0][0]}')
print(f'Total trans_ct: {output[0][1]}')
print(f'Total trans_amt: {output[0][2]}')
print(f'Total Interest: {output[0][3]}')
print(f'Total Annual Fees: {output[0][4]}')
print(f'Total Cust Acq Cost: {output[0][5]}')
print(f'Total Profit: {output[0][6]}')
```

```
Total Revenue: 55315545
Total trans_ct: 655651
Total trans_amt: 44522013
Total Interest: 7843382.230
Total Annual Fees: 2950015
Total Cust Acq Cost: 972936
Total Profit: 9820596
```

--> KPIs Calculation in cust_detail

[19]:
```
# Avg cust_satisfaction_score, credit_limit, customer_age,
# No of client_num, and income
query= """
    Select
    round(avg(cust_satisfaction_score), 2) as avg_CSS,
    round(avg(credit_limit), 2) as avg_credit_limit,
    round(avg(customer_age), 2) as avg_customer_age,
    round(avg(income), 2) as avg_income,
    count(distinct cust.client_num) as cust_count
     from cust_detail as cust
     join cc_detail as cc
     on cust.client_num= cc.client_num;
    """
output =execute_query(query)
print(f'Avg. Cust Statisfaction Score : {output[0][0]}')
print(f'Avg. Credit Limit: {output[0][1]}')
print(f'Avg. Customer Age: {output[0][2]}')
print(f'Avg. Income: {output[0][3]}')
print(f'Total Customers: {output[0][4]}')
```

```
Avg. Cust Statisfaction Score : 3.19
Avg. Credit Limit: 8635.64
Avg. Customer Age: 46.27
Avg. Income: 56976.10
Total Customers: 10108
```

Week On Week Revenue

SQL

--> Week on Week Revenue percent

[20]:
```
# CW, PW and WOW% caluclations
query= """
    with cte as
    (   Select weeknum2,
        sum(revenue) as current_week_revenue,
        lag(sum(revenue)) over(order by weeknum2) as prev_week_revenue
        from cc_detail
        group by weeknum2
    )
    select weeknum2, current_week_revenue, prev_week_revenue,
    round(((current_week_revenue-prev_week_revenue)/prev_week_revenue)*100, 2
        ) as WOW_Revenue_percent
    from cte
    order by weeknum2 desc
    """
wow_rev= pd.DataFrame(execute_query(query),
        columns =['weeknum', 'current_week_revenue', 'prev_week_revenue', 'WOW_Revenue%'])
wow_rev.head()
```

[20]:

	weeknum	current_week_revenue	prev_week_revenue	WOW_Revenue%
0	52	933137	1070444	-12.83
1	51	1070444	1026553	4.28
2	50	1026553	980160	4.73
3	49	980160	1008780	-2.84

POWER BI

```
Current Week revenue = CALCULATE(
    SUM('public cc_detail'[revenue]),
    FILTER(ALL('public cc_detail'),
        'public cc_detail'[WeekNum2]= max('public cc_detail'[WeekNum2])))
```

```
Previous week revenue = CALCULATE(
    sum('public cc_detail'[revenue]),
    FILTER(all('public cc_detail'),
    'public cc_detail'[WeekNum2] = max('public cc_detail'[WeekNum2])-1))
```

```
WOW Revenue = DIVIDE(([Current Week revenue]-[Previous week revenue]),
                    [Previous week revenue])
```

SQL Queries

--> Revenue By Card Category

[21]:
```
# Sum of revenue by each card category
query= """
    Select
    card_category,
    sum(revenue) as revenue
    from cc_detail
    group by card_category
    order by revenue desc
    """
rev_by_cardCat= pd.DataFrame(execute_query(query),
            columns =['card_category', 'revenue'])
rev_by_cardCat
```

[21]:

	card_category	revenue
0	Blue	46139521
1	Silver	5586343
2	Gold	2454073
3	Platinum	1135608

--> Revenue and Trans Count by qtr

[22]:
```
# Sum of revenue and trans Count by each qtr
query= """
    Select
    qtr,
    sum(revenue) as revenue,
    sum(total_trans_ct) as total_trans_ct
    from cc_detail
    group by qtr
    order by qtr
    """
rev_And_trn_ct_by_qtr= pd.DataFrame(execute_query(query),
            columns =['qtr', 'revenue', 'total_trans_ct'])
rev_And_trn_ct_by_qtr
```

[22]:

	qtr	revenue	total_trans_ct
0	Q1	13964401	163255
1	Q2	13820611	164201
2	Q3	14235481	166566
3	Q4	13295052	161629

--> Revenue by Exp Type

[23]:
```
# Sum of revenue by each exp type
query= """
    Select
    exp_type,
    sum(revenue) as revenue
    from cc_detail
    group by exp_type
    order by revenue desc
    """
rev_by_exptype= pd.DataFrame(execute_query(query),
            columns =['exp_type', 'revenue'])
rev_by_exptype
```

[23]:

	exp_type	revenue
0	Bills	13775147
1	Entertainment	9521170
2	Fuel	9327197
3	Grocery	8575828
4	Food	8250455
5	Travel	5865748

--> Revenue and by Customer Job

[24]:
```
# Sum of revenue by each Customer Job
query= """
    Select
    customer_job,
    sum(revenue) as revenue
    from cc_detail as cc
    join cust_detail as cust
    on cc.client_num = cust.client_num
    group by customer_job
    order by revenue desc
    """
rev_by_custJob= pd.DataFrame(execute_query(query),
            columns =['customer_job', 'revenue'])
rev_by_custJob
```

[24]:

	customer_job	revenue
0	Businessman	17387850
1	White-collar	10114685
2	Selfemployeed	8261781
3	Govt	8111727
4	Blue-collar	6904307
5	Retirees	4535195

--> Revenue and by Income Group

[25]:
```
# Sum of revenue by each exp type
query= """
    Select
    IncomeGroup,
    sum(cc.revenue) as revenue
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by IncomeGroup
    order by revenue desc
    """
rev_by_custJob= pd.DataFrame(execute_query(query),
            columns =['IncomeGroup', 'revenue'])
rev_by_custJob
```

[25]:

	IncomeGroup	revenue
0	High	29229968
1	Med	15854268
2	Low	9710957
3	Very Low	520352

REPORT- 1

Credit Card Transaction Report

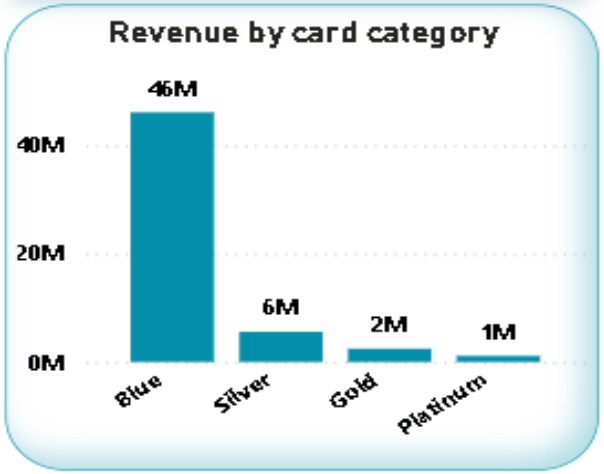
Q1 | Q2 | Q3 | Q4

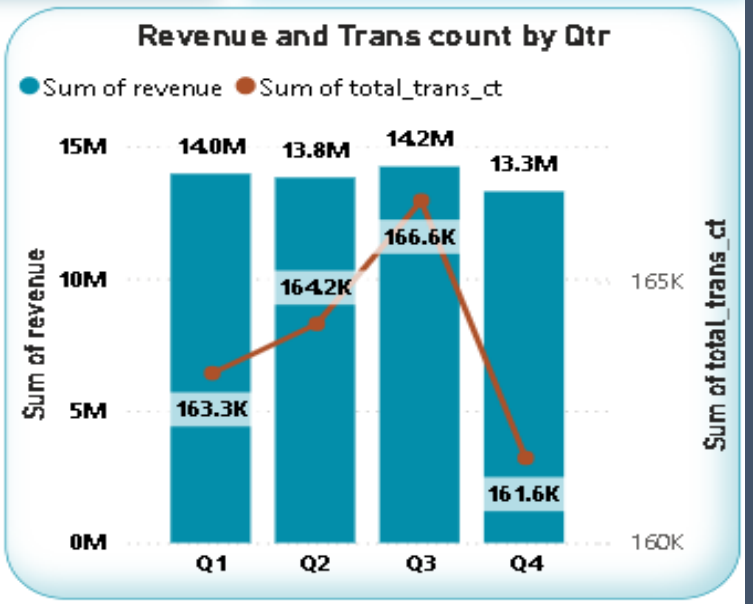
Week Start Date: All

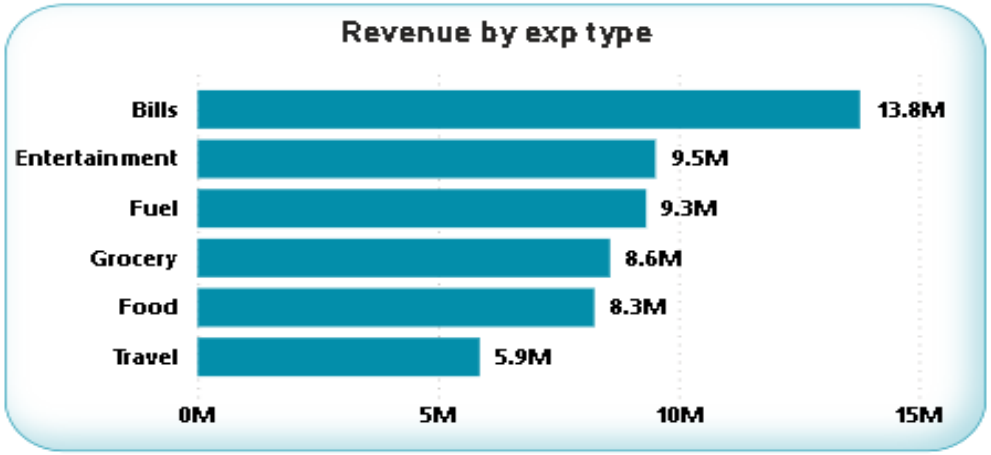
F | M

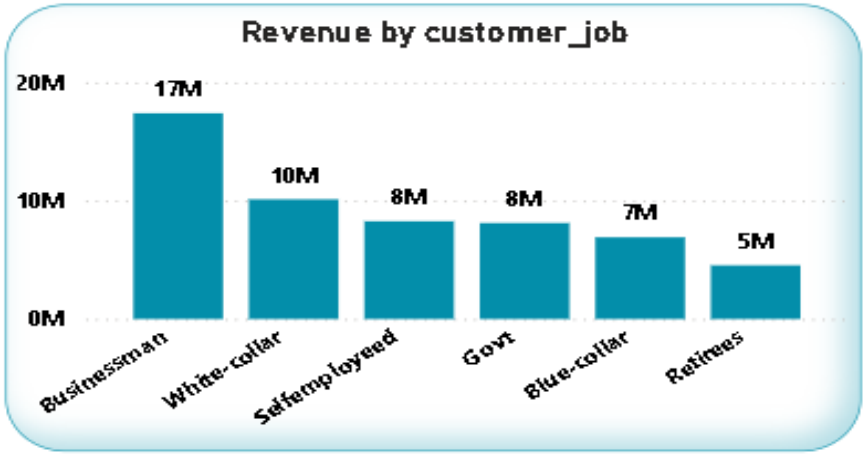
Revenue	Trans. Count	Trans. Amount	Interest
55M	656K	45M	8M

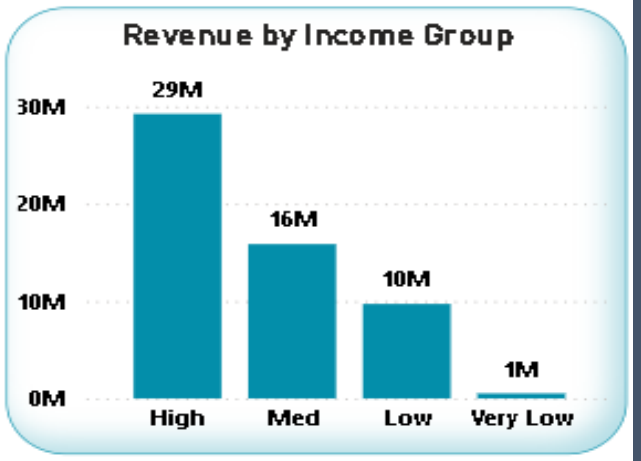
WeekNum2	Current Week revenue	Previous week revenue	WOW Revenue
52	9,33,137.00	10,70,444.00	-12.83%
51	10,70,444.00	10,26,553.00	4.28%
50	10,26,553.00	9,80,160.00	4.73%
49	9,80,160.00	10,08,780.00	-2.84%
48	10,08,780.00	10,47,116.00	-3.66%
47	10,47,116.00	10,78,921.00	-2.95%
46	10,78,921.00	10,94,932.00	-1.46%











REPORT - 1

KEY INSIGHTS

1. **Overall Revenue and Transactions**:
    - **Total Revenue**: 55 M
    - **Total Transaction Count**: 656 K
    - **Total Transaction Amount**: 45 M
    - **Total Interest**: 8 M

2. **Current Week (Week 52)** 0.9 M, a decrease from the previous week's 1 M (12.83% WoW revenue).

3. The **Blue Card** dominates revenue generation.

4. **Bills** are the **largest** revenue source, followed by **entertainment** and **fuel**.

5. **Businessmen** and **high-income** groups contribute significantly to revenue.

6. Revenue is **relatively stable** across **quarters**, with **Q3** slightly **leading** and **Q3** having the **highest** transaction count.

Percent Total

SQL

--> Percent of Total By Card Category

```
[30]: # percent total by card category
query= """
    with cte as (
        Select sum(revenue) as total_revenue
        from cc_detail
    )
    select card_category,
        sum(revenue) as revenue,
        (select * from cte) as total_revenue,
        round(sum(revenue)*100/(select * from cte), 2) as "%_total"
    from cc_detail
    group by card_category
    order by "%_total" desc
    """
per_total= pd.DataFrame(execute_query(query),
        columns =['card_category', 'revenue','total_revenue', '%_total'])
per_total
```

[30]:

	card_category	revenue	total_revenue	%_total
0	Blue	46139521	55315545	83.41
1	Silver	5586343	55315545	10.10
2	Gold	2454073	55315545	4.44
3	Platinum	1135608	55315545	2.05

POWER BI

```
1 TotalRevenue = CALCULATE(sum('public cc_detail'[revenue]), all('public cc_detail'))
2
```

```
1 Total% = DIVIDE(sum('public cc_detail'[revenue]), [TotalRevenue])
2
```

This can help to calculate the overall contribution of the each card category in total revenue.

--> Tran. Count by Week and Gender

[31]:
```
# Total Transaction count by Week and Gender
query= """
    Select
    week_start_date,
    Gender,
    count(cc.total_trans_ct) as total_trans_ct
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by week_start_date ,gender
    order by week_start_date
    """
trn_by_week_gender= pd.DataFrame(execute_query(query),
            columns =['week_start_date', 'Gender','total_trans_ct'])
trn_by_week_gender.head()
```

[31]:

	week_start_date	Gender	total_trans_ct
0	2023-01-01	F	125
1	2023-01-01	M	69
2	2023-01-08	M	71
3	2023-01-08	F	124
4	2023-01-15	F	112

--> Total Annual Fees by Card Category

[32]:
```
# Annual Fees by Card Category
query= """
    Select
    card_category,
    sum(annual_fees) as annual_fees
    from cc_detail
    group by card_category
    order by annual_fees desc
    """
afee_by_card_cat= pd.DataFrame(execute_query(query),
            columns =['card_category', 'annual_fees'])
afee_by_card_cat
```

[32]:

	card_category	annual_fees
0	Blue	2685635
1	Silver	187505
2	Gold	56210
3	Platinum	20665

--> Revenue by merital Status and Gender

[33]:
```
# Revenue by Merital Status and Gen
query= """
    Select
    marital_status,
    gender,
    sum(cc.revenue) as revenue
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by marital_status ,gender
    order by marital_status
    """
rev_by_marsts_gender= pd.DataFrame(execute_query(query),
            columns =['marital_status', 'Gender','revenue'])
rev_by_marsts_gender.head()
```

[33]:

	marital_status	Gender	revenue
0	Married	M	15415135
1	Married	F	12553039
2	Single	F	10805765
3	Single	M	12456344
4	Unknown	M	2350229

--> Revenue by Use Chip

[34]:
```
# revenue by use chip
query= """
    Select
    use_chip,
    sum(revenue) as revenue
    from cc_detail
    group by use_chip
    order by revenue desc
    """
rev_by_usechip= pd.DataFrame(execute_query(query),
            columns =['use_chip', 'revenue'])
rev_by_usechip
```

[34]:

	use_chip	revenue
0	Swipe	34912898
1	Chip	16966834
2	Online	3435813

--> Revenue and tran count Age Group

[35]:
```
# Revenue and tran Age Group
query= """
    Select
    AgeGroup,
    sum(cc.revenue) as revenue,
    sum(cc.total_trans_ct) as total_trans_ct
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by AgeGroup
    order by revenue desc
    """
rev_and_tran_by_agegrp= pd.DataFrame(execute_query(query),
            columns =['AgeGroup', 'revenue','total_trans_ct'])
rev_and_tran_by_agegrp
```

[35]:

	AgeGroup	revenue	total_trans_ct
0	40-50	24283550	293744
1	50-60	18188812	199738
2	30-40	9581920	117868
3	60+	2208952	31333
4	20-30	1052311	12968

REPORT - 2

Credit Card Transaction Report

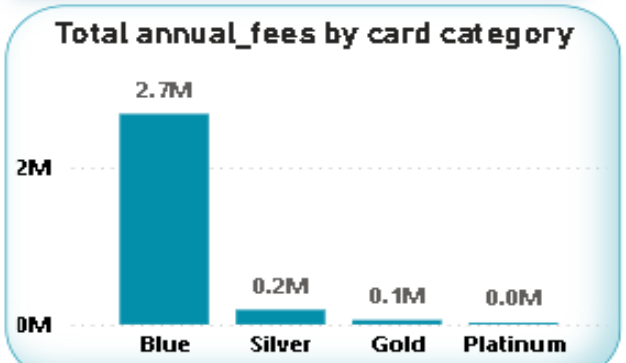
Q1 | Q2 | Q3 | Q4

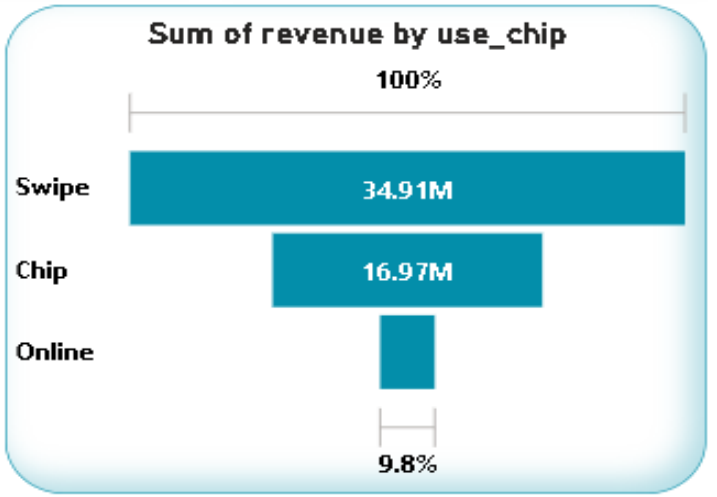
Week Start Date: All

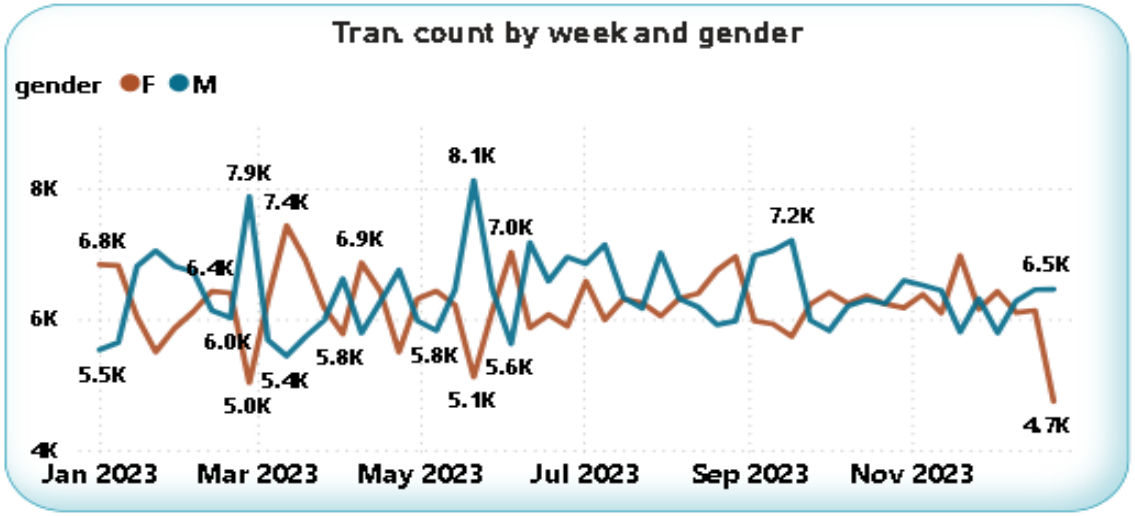
F | M

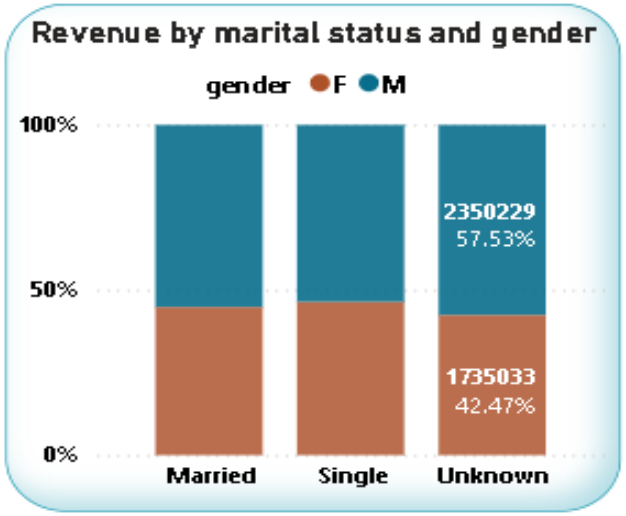
Revenue	Annual Fee	Acq. Cost	Profit
55M	3M	973K	10M

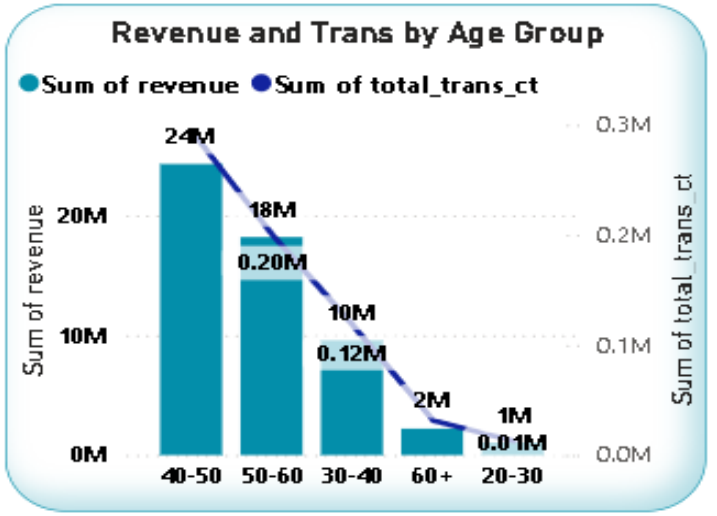
card_category	Sum of revenue	Total Revenue	Total %
Blue	4,61,39,521.00	55315545	83.41%
Silver	55,86,343.00	55315545	10.10%
Gold	24,54,073.00	55315545	4.44%
Platinum	11,35,608.00	55315545	2.05%











REPORT - 2

KEY INSIGHTS

1. **Overall Metrics**::
   - **Total Revenue**: 55 M
   - **Annual Fee Revenue**: 3M
   - **Acquisition Cost**: 973K
   - **Profit**: 10M

2. **Blue Card**: Dominates with **46.16M** revenue **(83.41%** of total).

3. The **Blue Card** is the **most significant** revenue generator in terms of **transaction amounts** and **annual fees.**

4. **Swipe** transactions dominate, followed by **chip** and **online** transactions.

5. **Females** tend to have **higher transaction counts** and contribute significantly to revenue, especially among **married** individuals.

6. The **40-50 age group** is the **largest** contributor to revenue, followed by the **50-60 age group**. **Younger** and **older age** groups contribute **less** to revenue and transactions.

--> Total Transaction amount by job and chip

[36]:
```
# tran amt by customer job and use chip
query= """
    Select
    customer_job,
    use_chip,
    sum(cc.total_trans_amt) as total_trans_amt
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by customer_job, use_chip
    order by customer_job asc,total_trans_amt desc
    """
tran_by_custjob_usechip= pd.DataFrame(execute_query(query),
             columns =['customer_job', 'use_chip','total_trans_ct'])
tran_by_custjob_usechip.head()
```

[36]:

	customer_job	use_chip	total_trans_ct
0	Blue-collar	Swipe	3808223
1	Blue-collar	Chip	1361027
2	Blue-collar	Online	319588
3	Businessman	Swipe	7120589
4	Businessman	Chip	6146576

--> No. of clinets by income group and delinquent_acc status

[37]:
```
# num of client by income grp and account status
query= """
    Select
    IncomeGroup,
    delinquent_acc,
    count(distinct cc.client_num) as client_num_count
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by IncomeGroup, delinquent_acc
    """
num_clnts_by_incmgrp_dlqnt_ac= pd.DataFrame(execute_query(query),
             columns =['IncomeGroup', 'delinquent_acc','client_num_count'])
num_clnts_by_incmgrp_dlqnt_ac.head(6)
```

[37]:

	IncomeGroup	delinquent_acc	client_num_count
0	High	0	2780
1	High	1	190
2	Low	0	3410
3	Low	1	219
4	Med	0	2974

--> Avg. Cust Satisfaction Score by income Grp

[38]:
```
# Avg CSS by income grp
query= """
    Select
    IncomeGroup,
    avg(cust_satisfaction_score) as css
    from cust_detail
    group by IncomeGroup
    order by css desc
    """
css_by_incmgrp= pd.DataFrame(execute_query(query),
             columns =['IncomeGroup', 'css'])
css_by_incmgrp.head(6)
```

[38]:

	IncomeGroup	css
0	Very Low	3.2144846796657382
1	Med	3.1965079365079365
2	Low	3.1928906034720309
3	High	3.1740740740740741

--> Avg. Credit limit by income grp

[39]:
```
# Avg. Credit Limit by income grp
query= """
    Select
    IncomeGroup,
    avg(cc.credit_limit) as credit_limit
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by IncomeGroup
    order by credit_limit desc
    """
avg_credit_limit_by_incmgrp= pd.DataFrame(execute_query(query),
             columns =['IncomeGroup', 'credit_limit'])
avg_credit_limit_by_incmgrp
```

[39]:

	IncomeGroup	credit_limit
0	High	9745.6584175084175084
1	Very Low	9568.2498607242339833
2	Low	8302.0242491044364839
3	Med	7867.1188253968253968

--> Avg. Credit limit by Age Grp

[40]:
```
# Avg. Credit Limit by Age grp
query= """
    Select
    AgeGroup,
    avg(cc.credit_limit) as credit_limit
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by AgeGroup
    order by credit_limit desc
    """
avg_credit_limit_by_agegrp= pd.DataFrame(execute_query(query),
             columns =['AgeGroup', 'credit_limit'])
avg_credit_limit_by_agegrp
```

[40]:

	AgeGroup	credit_limit
0	40-50	8680.2187017001545595
1	50-60	8639.3596391580354160
2	30-40	8598.5288200108754758
3	20-30	8560.3116279069767442
4	60+	8393.9889097744360902

--> Avg Income by Age Grp

[41]:
```
# Avg. Credit Limit by Age grp
query= """
    Select
    AgeGroup,
    avg(income) as income
    from cust_detail
    group by AgeGroup
    order by income desc
    """
avg_income_by_agegrp= pd.DataFrame(execute_query(query),
             columns =['AgeGroup', 'income'])
avg_income_by_agegrp
```

[41]:

	AgeGroup	income
0	50-60	63422.867023053792
1	40-50	56243.112166041069
2	30-40	54019.847743338771
3	20-30	47568.465116279070
4	60+	40968.088345864662

REPORT - 3

Credit Card Customer Report

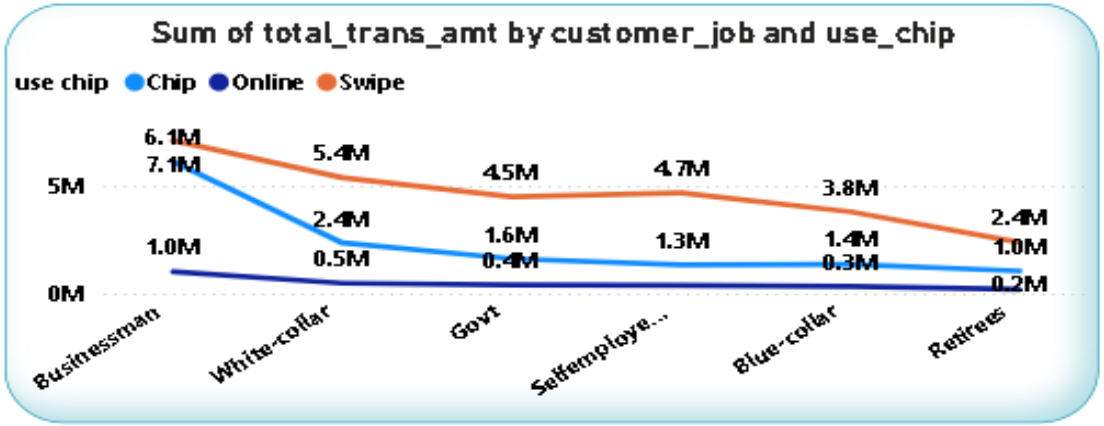
Q1	Q2	Q3	Q4

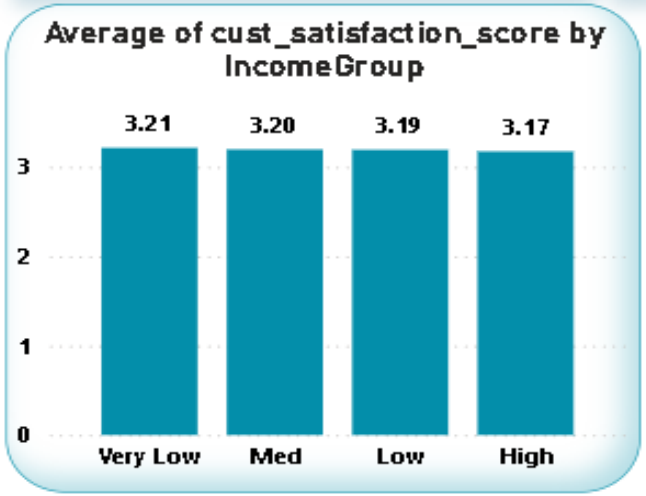
Week Start Date: All

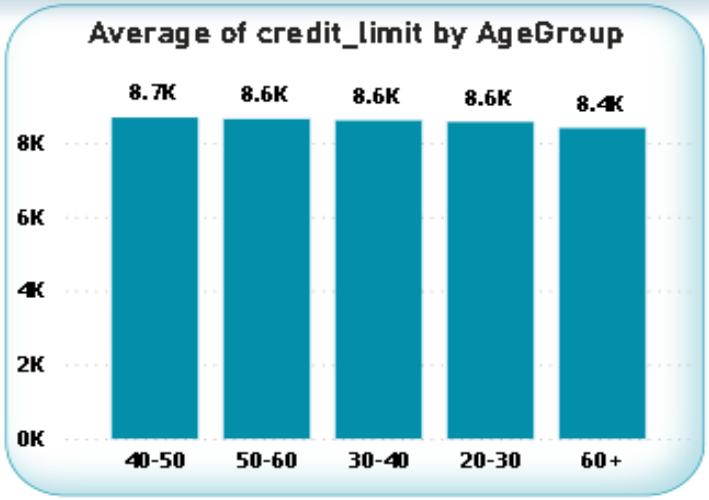
F	M

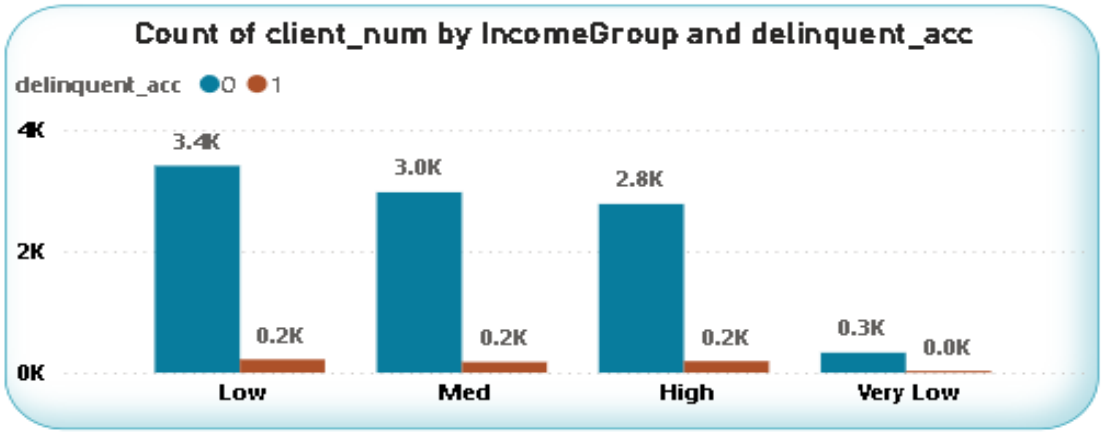
Married	Single	Unknown

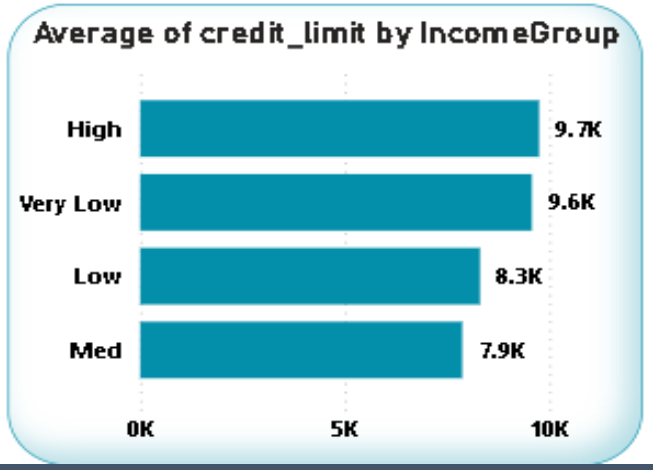
Revenue	Avg. Satis Score	Avg. Credit Limit	Avg. Age
55M	3.19	8.64K	46.27

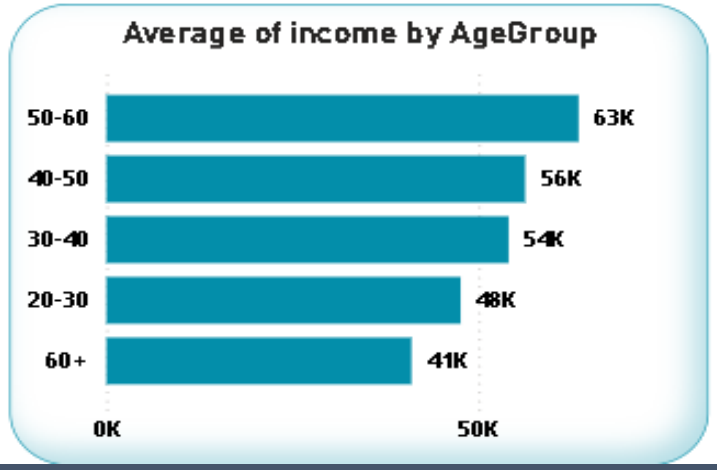












REPORT - 3

KEY INSIGHTS

1. Overall Metrics::

- **Total Revenue**: 55 M
- **Average Satisfaction Score**: 3.19
- **Average Credit Limit**: 8.64K
- **Average Age**: 46.27

2. **Businessmen** Prefer using **chips** with a **total transaction amount of 7M,** followed **by swipe (6M)** and **online (1M).**

3. **Very Low-Income** group has the Highest average **satisfaction score of 3.21**.

4. **40-50** group has the highest **average credit limit of 8.7K**.

5. **Low-Income** group has **3.4K clients** with **0.2K delinquent accounts**.

6. **High Income** has the highest **average credit limit of 9.7K**.

7. **50-60** group Highest **average income of 63K**..

-->No client and percent by deliquent acc status

[42]:
```
# No client by deliquent acc status and percent
query= """
    with cte as (
    select count(distinct client_num) as total_cust
    from cc_detail
    )

    Select
      delinquent_acc,
      count( distinct client_num) as no_cust,
      (count( distinct client_num)*100/(select * from cte))
      as per_total
    from cc_detail
    group by delinquent_acc
    order by no_cust desc
    """
client_by_dlqntac= pd.DataFrame(execute_query(query),
           columns =['delinquent_acc', 'no_cust', 'per_total'])
client_by_dlqntac
```

[42]:

	delinquent_acc	no_cust	per_total
0	0	9494	93
1	1	614	6

--> Trans count by Age Grp and house owners

[43]:
```
# tran count by age and house owner
query= """
    Select
    AgeGroup,
    house_owner,
    sum(cc.total_trans_ct) as total_trans_ct
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by AgeGroup, house_owner
    order by total_trans_ct desc
    """
tran_cnt_by_agegrp_hsownr= pd.DataFrame(execute_query(query),
           columns =['AgeGroup', 'house_owner','credit_limit'])
tran_cnt_by_agegrp_hsownr.head()
```

[43]:

	AgeGroup	house_owner	credit_limit
0	40-50	no	154890
1	40-50	yes	138854
2	50-60	no	104793
3	50-60	yes	94945
4	30-40	no	66112

--> Cust Satisfaction Score by income grp

[44]:
```
# css by income grp
query= """
    Select
    AgeGroup,
    avg(cust_satisfaction_score) as css
    from cust_detail
    group by AgeGroup
    order by css desc
    """
css_by_incomegrp= pd.DataFrame(execute_query(query),
           columns =['AgeGroup', 'css'])
css_by_incomegrp
```

[44]:

	AgeGroup	css
0	60+	3.2669172932330827
1	30-40	3.2060902664491572
2	20-30	3.1953488372093023
3	50-60	3.1941196124290010
4	40-50	3.1697946566570987

--> Trans count By agegrp and personal load

[45]:
```
# tran count by age and personal loan
query= """
    Select
    AgeGroup,
    personal_loan,
    sum(cc.total_trans_ct) as total_trans_ct
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by AgeGroup, personal_loan
    order by total_trans_ct desc
    """
tran_cnt_by_agegrp_prloan= pd.DataFrame(execute_query(query),
           columns =['AgeGroup', 'personal_loan','total_trans_ct'])
tran_cnt_by_agegrp_prloan.head()
```

[45]:

	AgeGroup	personal_loan	total_trans_ct
0	40-50	no	256456
1	50-60	no	172922
2	30-40	no	104532
3	40-50	yes	37288
4	60+	no	27409

--> Trans count by age grp and car owner

[46]:
```
# tran count by age and car owner
query= """
    Select
    AgeGroup,
    car_owner,
    sum(cc.total_trans_ct) as total_trans_ct
    from cust_detail as cust
    join cc_detail as cc
    on cust.client_num = cc.client_num
    group by AgeGroup, car_owner
    order by total_trans_ct desc
    """
tran_cnt_by_agegrp_carownr= pd.DataFrame(execute_query(query),
           columns =['AgeGroup', 'car_owner','total_trans_ct'])
tran_cnt_by_agegrp_carownr.head()
```

[46]:

	AgeGroup	car_owner	total_trans_ct
0	40-50	no	174745
1	50-60	no	123353
2	40-50	yes	118999
3	50-60	yes	76385
4	30-40	no	68570

REPORT - 4

Credit Card Customer Report

Q1	Q2	Q3	Q4

Week Start Date: All

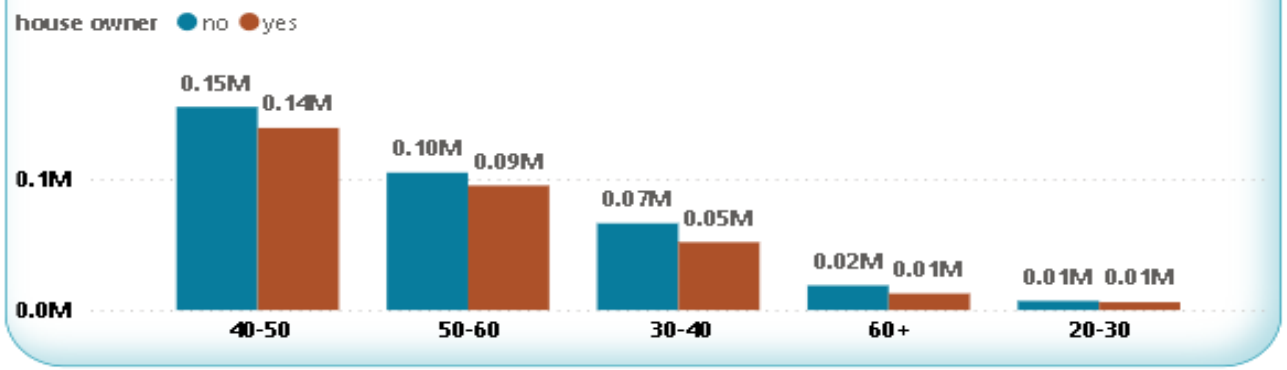
High
Low
Med
Very Low

F	M

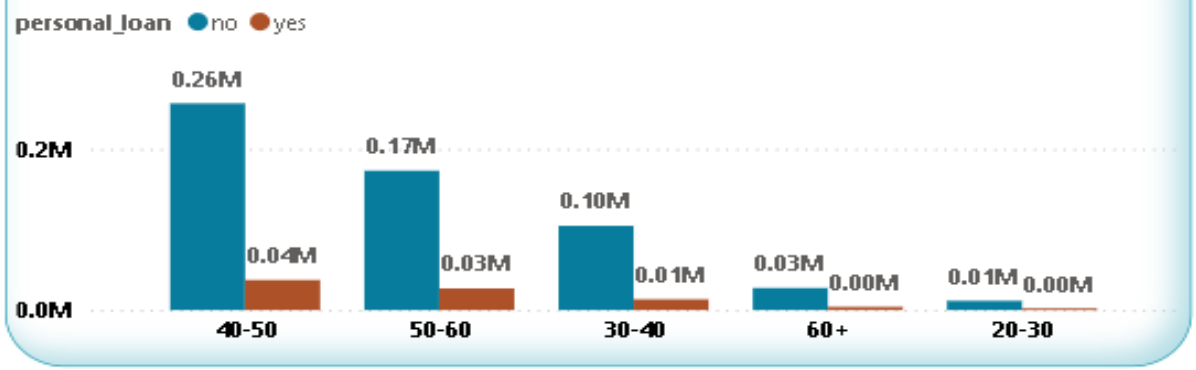
Married	Single	Unknown

Revenue	Cust Count	Avg. Income
55M	10.11K	56.98K

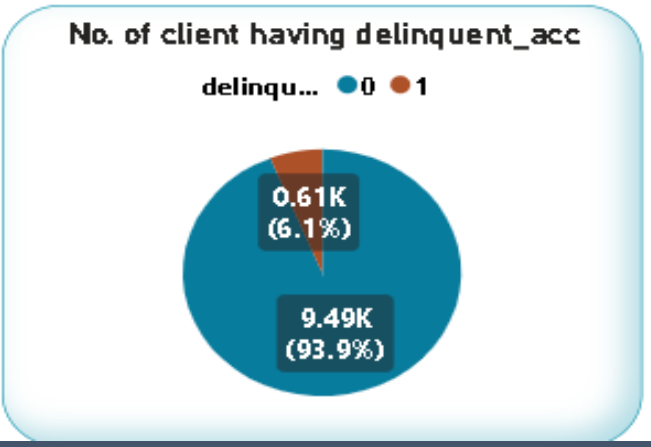
Trans. by Age grp and House owner



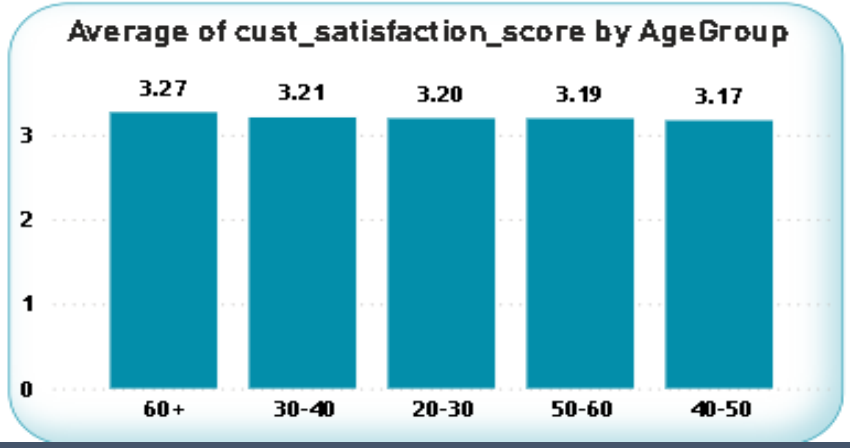
Sum of total_trans_ct by AgeGroup and personal_loan



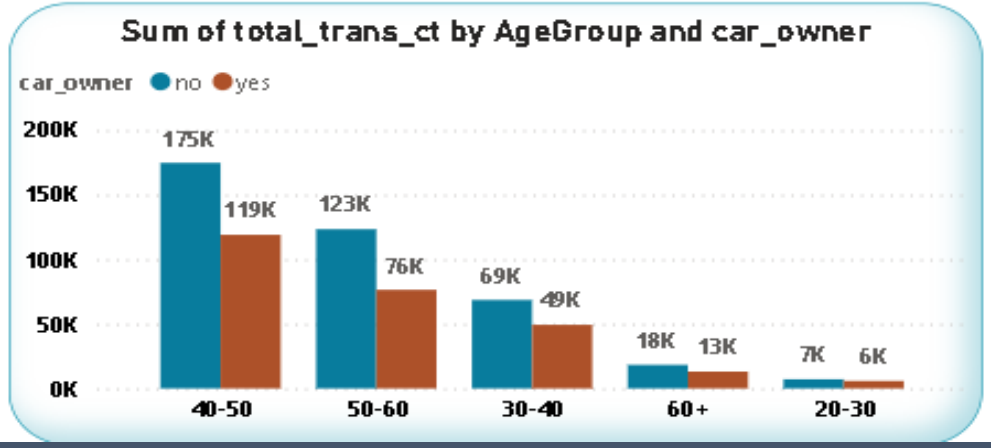
No. of client having delinquent_acc



Average of cust_satisfaction_score by AgeGroup



Sum of total_trans_ct by AgeGroup and car_owner



REPORT - 4

KEY INSIGHTS

1. **Overall Metrics**::
   - **Total Revenue**: 55 M
   - **Customer Count**: 10.11K
   - **Average Income**: 56.98K.

2. **Age Group 40-50 Non-house owners** have higher **transaction amounts (0.15M)** compared to **house owners (0.14M).**

3. **Age Group 40-50** Customers **without personal loans** have significantly higher **transaction counts (0.26M)** compared to those **with personal loans (0.04M).**

4. **Delinquent Accounts 0.61K (6.1%)** of clients have **delinquent accounts**.

5. **Age Group 60+** Highest average **customer satisfaction score of 3.27**.

6. **Age Group 40-50 Car owners** have **higher transaction counts (175K)** compared to **non-car owners (119K).**

FINAL PROJECT LINKS

DRIVE LINK : https://drive.google.com/drive/folders/1vq7WyUJQd5DK9zbQv-2q95gbh5yzJEXr?usp=drive_link

GITHUB LINK: https://github.com/KshmaKshma/Credit_Card_Financial_Analysis

THANK YOU