

Optimizing Marketing Strategies of Islamic Banks

By: Haaniyah Muhammad Mundia (14804)

&

Hurma Mehmood (14855)

Introduction:

As per our business knowledge template we found out some of the metrics in order to optimize the marketing strategies of Islamic Banking products and services. Those are:

- Customer Life - how long does the bank retain the customer.
- Referral Marketing - how much the bank is being referred to others
- Customer Satisfaction Index - how satisfied customers are with the bank's products and services.

It can be seen that these metrics are either generating demand or harvesting demand hence the bank should plan out its campaigns such that it fulfills both kinds of demands.

For the purpose of this project we would be following the 2nd track, hence would analyze the data to optimize Islamic bank's marketing campaigns on HIVE.

Component 2 - ERD:

As submitted in the project progress document, our ERD is as follows:

Component 5 - Data Generation:

For the purpose of this project we generated data mostly via python's library Faker, while some were created manually. The link to the code is: ([add link](#))

Data for the Customer table was generated as follows:

- 5000 records were generated using the for loop(5000 rows were selected since our laptops could not process more data)

Customer Table:

- This table was generated in two halves. First half was generated using the profile() method which from which we used columns, ssn, name, sex, address, mail and birthdate. The rest of the columns were dropped.

- The columns, customer ID, Account ID, Login ID, City, State, Phone Number, and mobile number were generated using random.randint function. Here account ID consists of 12 digits since according to our research an account ID is usually 8-12 digits long. Similarly, login ID varies from 8-32 characters so we have taken 8 digits in login ID. The length of mobile and phone number is 10 because phone and mobile numbers in the US are of the same length.

In [130]:

```
customers = []
for i in range(5000):
    data = []
    cust_id = i+1
    data.append(cust_id)
    acc_id = fake.random_number(digits=12)
    data.append(acc_id)
    login_id = fake.random_number(digits=8)
    data.append(login_id)
    city = fake.city()
    data.append(city)
    state = fake.state()
    data.append(state)
    phone_no = fake.random_number(digits=10)
    data.append(phone_no)
    mobile_no = fake.random_number(digits=10)
    data.append(mobile_no)

    customers.append(data)
```

In [221]:

```
df = pd.DataFrame(customers, columns=['Customer ID', 'Account ID', 'Login ID', 'City', 'State', 'Phone Number', 'Mobile Number'])
df.head()
```

Out[221]:

	Customer ID	Account ID	Login ID	City	State	Phone Number	Mobile Number
0	1	875869972702	40293236	Lake Kathleenfort	Vermont	6622094774	7503444179
1	2	894958903051	74378334	North Laurenfurt	Maine	5682730469	2550284723
2	3	823386251343	61057234	Richardland	Georgia	6169457658	9302182877
3	4	660229224253	10603674	West Benjamin	Delaware	8490513222	9761896396
4	5	272832269257	83775861	North Jane	Pennsylvania	6442533160	2121329855

- Then we concatenated these two dataframes into one using pandas function ‘concat’ and then saved them into the Customer.csv using ‘to_csv’ function

In [225]:

```
customer_df = pd.concat([df, CustomerProfile], 1)
customer_df.head()
```

Out[225]:

	Customer ID	Account ID	Login ID	City	State	Phone Number	Mobile Number	job	company	ssn	residence	current_loc
0	1	875869972702	40293236	Lake Kathleenfort	Vermont	6622094774	7503444179	Electronics engineer	Avila Ltd	641-96-5526	474 Jonathan PortInFreemanberg, NC 76646	(-79.56,-73.18)
1	2	894958903051	74378334	North Laurenfurt	Maine	5682730469	2550284723	Farm manager	Guzman LLC	033-90-1776	83982 Eugene MountInDavidborough, MD 60330	(44.55,59.96)
2	3	823386251343	61057234	Richardland	Georgia	6169457658	9302182877	Physicist, medical	Jones, Foster and Taylor	568-74-6433	583 Derrick CapeInSouth Cassandraport, TX 22595	(-81.273,-121.11)
3	4	660229224253	10603674	West Benjamin	Delaware	8490513222	9761896396	Air broker	Powell, Downs and Robinson	631-70-0186	186 Devon Islands Suite 218InSouth Shelbytown,...	(9.453,10.74)
4	5	272832269257	83775861	North Jane	Pennsylvania	6442533160	2121329855	Sales	Burns	458-97	807 Taylor Square Apt. 197InPort Isamine	(78.764,-)

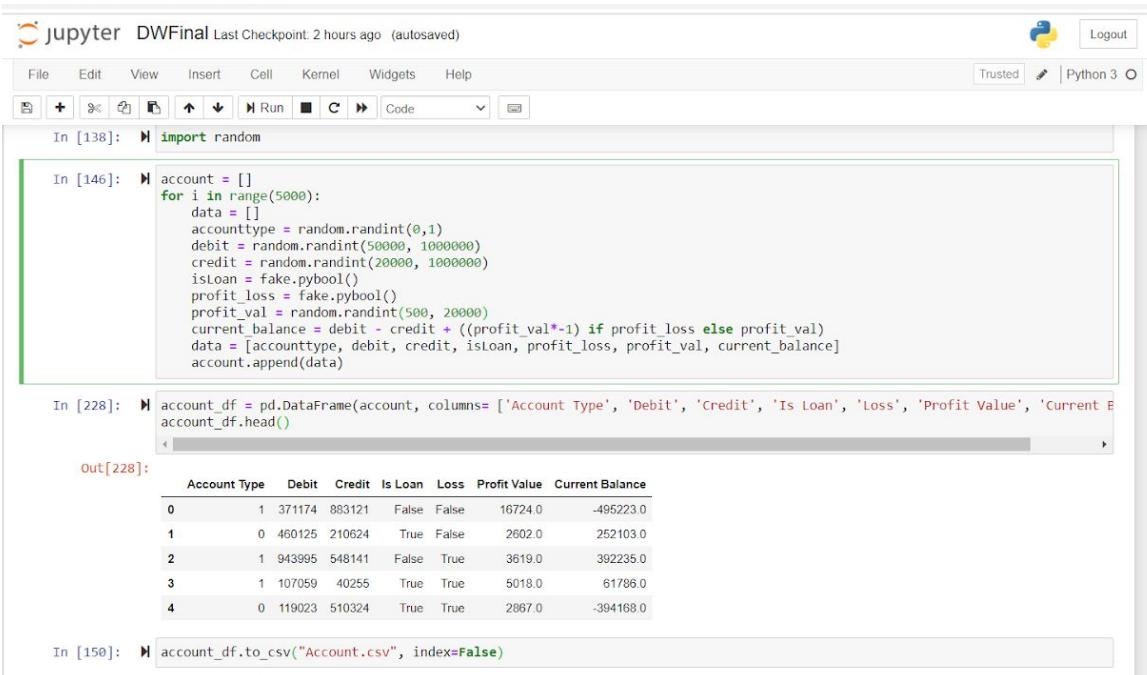
In [136]:

```
customer_df.to_csv("Customer.csv", index=False)
```

Account Table:

- 5000 records were generated using the for loop
- Here column ‘account type’ has two value 0 or 1 which represent 0=Current and 1=Savings

- The ‘isLoan’ column was created using the ‘pybool’ function which contains boolean values true and false which means if a user has taken loan then it’s true else false. Similarly the ‘Loss’ also has boolean values where if a the user has suffered loss then it is True else False
- The ‘Current Balance’ column was created by subtracting the credit column from debit and then subtracting the ‘Profit Value’ column if the ‘Loss’ is True else we’ve added the Profit Value.
- Then this was saved into ‘Account.csv’



The screenshot shows a Jupyter Notebook interface with the following details:

- Header:** jupyter DWFinal Last Checkpoint: 2 hours ago (autosaved), Logout
- Toolbar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help
- In [138]:** import random
- In [146]:** account = []
for i in range(5000):
 data = []
 accounttype = random.randint(0,1)
 debit = random.randint(50000, 1000000)
 credit = random.randint(20000, 1000000)
 isloan = fake.pybool()
 profit_loss = fake.pybool()
 profit_val = random.randint(500, 20000)
 current_balance = debit - credit + ((profit_val*-1) if profit_loss else profit_val)
 data = [accounttype, debit, credit, isloan, profit_loss, profit_val, current_balance]
 account.append(data)
- In [228]:** account_df = pd.DataFrame(account, columns= ['Account Type', 'Debit', 'Credit', 'Is Loan', 'Loss', 'Profit Value', 'Current Balance'])
account_df.head()
- Out[228]:**

	Account Type	Debit	Credit	Is Loan	Loss	Profit Value	Current Balance
0	1	371174	883121	False	False	16724.0	-495223.0
1	0	460125	210624	True	False	2602.0	252103.0
2	1	943995	548141	False	True	3619.0	392235.0
3	1	107059	40255	True	True	5018.0	61786.0
4	0	119023	510324	True	True	2867.0	-394168.0

- In [150]:** account_df.to_csv("Account.csv", index=False)

- Here account open date and close date were also added to the ‘Account.csv’ and then dates from the ‘Closed date’ column were randomly removed.



The screenshot shows a Jupyter Notebook interface with the following details:

- In [220]:** from datetime import timedelta
account_date = []
for i in range(5000):
 data = []
 opendate = fake.date_between(start_date='-33y', end_date='today')
 closeddate = opendate + timedelta(days=random.randint(1,10000))
 data = [opendate, closeddate]
 account_date.append(data)
account_date_df = pd.DataFrame(account_date, columns=['Open Date', 'Closed Date'])
account_date_df.to_csv("DateAccount.csv", index=False)

User Login Account Table:

- 5000 records were generated using the for loop
- ‘User security pin’ contains 4 digit pin which were randomly assigned
- ‘User password’ was created using the ‘.password’ function
- This was then saved to ‘UserLoginAccount.csv’ and Login ID was manually taken from the customer table.

```
In [159]: M user_login = []
for i in range(5000):
    data = []
    pin = random.randint(1001,9999)
    password = fake.password()
    data = [pin, password]
    user_login.append(data)

In [160]: M user_login_df = pd.DataFrame(user_login, columns=['User Security Pin', 'User Password'])

In [229]: M user_login_df.head()

Out[229]:
   User Security Pin  User Password
0              3454  PEIbSYza@#
1              1426  &5Xz+XaIEB
2              1335  IBMhE2pyP^
3              9619  %)fpJ2KsHa
4              7868  o8dnTbyP1
```

Transaction Log Table:

- 5000 records were generated using the for loop
- ‘Transaction ID’ was created by incrementing in the loop
- Random transactions were created on the basis of randomly generating customer ID.
- Since columns ‘Login ID’ and ‘Account ID’ were also needed in this table, with the help of pandas ‘iloc’ function the corresponding customers row was fetched which allows us to select their corresponding account and login id.
- Similarly ‘Current balance’ was fetched from the ‘account_df’ dataframe.
- This was then saved to ‘Transaction_log.csv’

```
In [178]: M transaction_log = []
for i in range(10000):
    data = []
    transaction_id = 100 + i
    cus_id = random.randint(1,5000)
    acc_id = customer_df.iloc[cus_id-1]['Account ID']
    login_id = customer_df.iloc[cus_id-1]['Login ID']
    transtype_id = random.randint(1,8)
    current_bal= account_df.iloc[cus_id-1]['Current Balance']
    trans_amount = random.randint(100,50000)
    upd_amount = current_bal - trans_amount
    trans_date = fake.date_between(start_date='-15y', end_date='today')
    data = [transaction_id,cus_id, acc_id, login_id, transtype_id, current_bal, trans_amount, upd_amount, trans_date]
    transaction_log.append(data)

In [179]: M transaction_log_df = pd.DataFrame(transaction_log, columns=['Transaction ID', 'Customer ID', 'Account ID', 'Login ID', 'Trans
        <      >
        In [180]: M transaction_log_df.to_csv("Transaction_log.csv", index=False)
```

LinkedIn, Twitter, Youtube, Facebook Table:

- 2000 records were generated using the for loop since we are only using the transactional log for the last 15 years and according to our research at least 2 ads/posts per week are made by a company therefore we’ve taken 2000 data
- Here all the columns are generated randomly. Similarly twitter, linkedIn and youtube tables were created but youtube has half(1000) records since youtube has video commercials which are created very often.

- The rest of the tables were created manually. The following are the details of what logic was used behind them:

Social Media Table:

- This table is just a junction table which consists of foreign keys ‘Facebook ID’, ‘Youtube ID’, ‘LinkedIn’, ‘Twitter ID’

Campaign Table:

- After extensively researching campaign ideas for islamic banks, we've selected the following 8 campaigns for this bank.
- The ‘Campaign Status’ tells us the status of the campaign where eligible means the campaign is running, pending means the campaign still awaits approval and paused means the campaign is for the time being.
- The ‘Campaign Contract’ tells us that the ‘agree’ means that agreement has been confirmed with the said party
- The ‘Campaign COD’ is the code for the corresponding campaign

Advertising Table:

- The advertising table consists of foreign keys ‘Campaign ID’ and ‘Social Media ID’.
- ‘Description’ tells us the description of the ad which we have kept the same as the campaign name.
- The ‘Costs’ were roughly calculated on the basis of our marketing knowledge. These numbers represent the amount it takes for one ad/post to run.
- ‘Type’ displays the type of Advertisement it was. Here, we’ve mainly focused on display ads since display ads can be run on any social media platform/website
- The ‘Start Date’ and ‘End/Pause Date’ represent the start date of the advertisement and the date date which it was ended/paused respectively.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Advertising Campaign	Social Media ID	Description	Costs	Type	Start Date	End/Pause Date						
2	1	1	10 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
3	2	1	11 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
4	3	1	114 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
5	4	1	115 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
6	5	1	218 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
7	6	1	219 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
8	7	1	332 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
9	8	1	333 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
10	9	1	426 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
11	10	1	427 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
12	11	1	530 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
13	12	1	531 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
14	13	1	634 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
15	14	1	738 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
16	15	1	738 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
17	16	1	824 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
18	17	1	1200 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
19	18	1	1201 Best Bank in Pakistan	2760	Display Ad	11/01/2010							
20	21	1	2760 Best Bank in Pakistan	2760	Display Ad	11/01/2010							

Product Table:

- The ‘Product Name’ were also selected based on research on the types of product Islamic Bank offers.
- ‘Product Description’ tells us a brief description of the product.
- ‘Product Charges’ were roughly calculated after extensive research on the products charges of famous banks.
- The ‘Start Time’ and ‘End Time’ display the date on which this product was launched and the date till which it is still going

A	B	C	D	E	F	G	H	I	J	K	L
1	Product ID	Product Name	Product Description	Product CI	Start Time	End Time					
2	81	Bachat Account	Bachat Account is a savings	500	10/01/1991	11/01/2001					
3	82	Current Account	This is a checking account t	500	11/01/1987	12/01/2021					
4	83	Savings Account	Savings Account provides n	500	12/01/1987	13/01/2021					
5	84	Karobarī Munāfiyah Account	Karobarī Munāfiyah Account i	500	13/01/1998	14/01/2021					
5	85	Mudarabah Certificate	Mudarabah Certificate is a	350	14/01/2000	15/01/2021					
7	86	Certificates of Islamic Investment	Certificates of Islamic Invest	350	15/01/1999	16/01/2021					
3	87	Smart Remittance Wallet	Smart Remittance Wallet is	550	16/01/2000	17/01/2021					
8	0										
9	1										
10	2										
11	3										
12	4										
13	5										
14	6										
15	7										
16	8										
17	9										

Services Table:

- The ‘Service Name’ were also selected based on research on the types of services Islamic Bank offers.
- ‘Service Description’ tells us a brief description of the service.

- ‘Service Charges’ were roughly calculated after extensive research on the services charges of famous banks.
- The ‘Start Time’ and ‘End Time’ display the date on which this service was started and the date till which it is still going

A	B	C	D	E	F	G
Service ID	Service Name	Service Description	Service Ch	Start Time	End Time	
101	Mobile Banking App	The App allows customers to check	75	11/01/1992	11/03/2023	
102	Labbalk	Labbalk facilitates customers who v	250	12/01/1999	12/03/2023	
103	SMS Banking	SMS Banking is an interactive servic	20	13/12/1992	13/03/2023	
104	Kafalah Plan	Kafalah is a savings plan with Takaf	150	14/01/1999	14/03/2023	
105	Takaful	The bank is offering Takaful covera	150	15/01/1995	15/03/2023	
106	Internet Banking	Internet Banking offers a suite of fe	35	16/01/1996	16/03/2023	

- **ServicesProduct:**

- This table is just a junction table which consists of foreign keys ‘Product ID’, ‘Services ID’, ‘Campaign ID’

A	B	C	D	E	F	G	H	I	J	K	L	M	N
SP ID	Product ID	Services ID	Campaign ID										
2	501	81	2										
3	502	82	2										
4	503	83	2										
5	504	84	8										
6	505	85	8										
7	506	86	8										
8	507	87	3										
9	508		101	3									
10	509		102	7									
11	510		103	1									
12	511		104	7									
13	512		105	8									
14	513		106	1									
15	514		101	1									
16	515		106	3									
17	516		101	5									
18	517		106	5									
19	518	81		6									

Component 6: Dimensional Modeling with Hive

For this purpose we have uploaded all the tables on Hive.

- First we’ve created a database ‘IslamicBank’

```
@quickstart:/ 
File Edit View Search Terminal Help
)
    at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:691)
    at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:626)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.
java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcces
sorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at org.apache.hadoop.util.RunJar.run(RunJar.java:221)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:136)
FAILED: ParseException line 1:5 cannot recognize input near 'show' 'database' 's
how' in ddl statement
hive> show databases;
OK
default
Time taken: 0.289 seconds, Fetched: 1 row(s)
hive> create database Islamic Bank;
FAILED: ParseException line 1:24 extraneous input 'Bank' expecting EOF near '<E0
F>''
hive> create database IslamicBank;
OK
Time taken: 0.187 seconds
hive> 
```

- Then we added all the tables in Hive, here is the step by step procedure of creating the product table, all the other tables are added similarly.

Databases > islamicbank > Create a new table from a file

Step 1: Choose File Step 2: Choose Delimiter Step 3: Define Columns

Choose a Delimiter

Beeswax has determined that this file is delimited by commas.

Delimiter: Comma (,)

Table preview:

col_1	col_2	col_3	col_4	col_5	col_6
Product ID	Product Name	Product Description	Product Charges	Start Time	End Time
81	Bachat Account	Bachat Account is a savin...	500	10/01/1991	11/01/2001
82	Current Account	This is a checking accoun...	500	11/01/1987	12/01/2021
83	Savings Account	Savings Account provides ...	500	12/01/1987	13/01/2021
84	Karobari Munafah Account	Karobari Munafah Account i...	500	13/01/1990	14/01/2021
85	Mudarabah Certificate	Mudarabah Certificate is ...	350	14/01/2000	15/01/2021
86	Certificates of Islamic I...	Certificates of Islamic I...	350	15/01/1999	16/01/2021
87	Smart Remittance Wallet	Smart Remittance Wallet i...	550	16/01/2000	17/01/2021

- Then we've created two fact table which are as follows:
 1. 'Customers_Fact' - one row of the fact table represents one customer
 2. 'Campaign_Fact' - one row of the fact table represents one campaign

- **Query 1:**

This query is for Customer_Fact in which we calculate the customer's life with the bank or the number of customers that left the bank for any reason. Moreover it has the date condition i.e. '2010-01-01' which marks the beginning of digital marketing campaigns by the bank.

```
1 select count(*) from account where closeddate < '2010-01-01';

Recent queries Query Log Columns Results Chart
1 _e0
1 280

1 select count(*) from account where closeddate >= '2010-01-01';

Execute Save Save as... Explain Format or create a New query
```

	Recent queries	Query	Log	Columns	Results	Chart
1	280				38	

It can be seen that before digital advertisements started the number of customers lost were significantly more than that after digital advertisement i.e. 280 vs 38.

It can be evaluated that digital advertisements helped improve customer life.

- **Query 2:**

This query is also for Customer_Fact and calculates the average customer life of customers before and after digital advertisement

	2923.275
1	select avg(Datediff(closeddate, opendate)) as CLV from account 2 where closeddate >= '2010-01-01' AND Datediff(closeddate, opendate) is not null;

Cancel Save Save as... Format or create a New query

	4823.957
1	

It can be seen that once again before digital advertisement the average customer life was way less than that after digital advertisement i.e. around 2923 days as compared to 4823 days.

This implies that digital marketing plays a significant role in improving average customer life.

- **Query 3:**

This query is for the Campaign_Fact where we get to know how which social media generates the most leads on average hence is a better platform to run campaigns in future.

The screenshot shows four separate queries being run in a MySQL Workbench interface. Each query uses an INNER JOIN between the 'facebook', 'linkedin', 'twitter', and 'youtube' tables and their corresponding 'socialmedia' tables based on the 'facebookid', 'linkedinid', 'twitterid', and 'youtubeid' columns respectively. The goal is to find the average number of clicks for each platform.

Facebook Query:

```
1 SELECT avg(facebook.clicks) FROM facebook
2 INNER JOIN socialmedia ON facebook.facebookid = socialmedia.facebookid;
```

LinkedIn Query:

```
1 SELECT avg(linkedin.clicks) FROM linkedin
2 INNER JOIN socialmedia ON linkedin.linkednid = socialmedia.linkednid;
```

Twitter Query:

```
1 SELECT avg(twitter.clicks) FROM twitter
2 INNER JOIN socialmedia ON twitter.twitterid = socialmedia.twitterid;
```

YouTube Query:

```
1 SELECT avg(youtube.clicks) FROM youtube
2 INNER JOIN socialmedia ON youtube.youtubeid = socialmedia.youtubeid;
```

Recent queries	Query	Log	Columns	Results	Chart
	_c0			1 25075.15	

It can be seen that facebook is the best platform to run a campaign as it has the highest average conversion as compared to all other social media platforms i.e. 25678 clicks. While linkedin has 25021, youtube has 25075 and twitter has 24587 average clicks which shows that linkedin is an equally good social media platform as Facebook in generating leads. Moreover, marketing strategies of twitter and youtube should be further looked upon in order to improve their leads.

- **Query 4:**

This query is for the Campaign_Fact table as it measures the customers satisfaction level and eventually tells us promoters i.e. people who speak good about us (mentioned in business knowledge template) for our bank hence improving referral marketing.

For this purpose we used two platforms i.e. Facebook and Youtube. Satisfied customers / promoters were those who liked, heart reacted or laughed while dissatisfied customers / detractors were those who sad or angry reacted on Facebook. While promoters on Youtube were those who liked and detractor those who disliked. The queries and results are as follows:

1 select sum(likes), sum(hearts), sum(laugh), sum(sad), sum(angry) from facebook;
<input type="button" value="Cancel"/> <input type="button" value="Save"/> <input type="button" value="Save as..."/> <input type="button" value="Format"/> or create a <input type="button" value="New query"/>
1 select sum(likes), sum(dislikes) from youtube;
<input type="button" value="Cancel"/> <input type="button" value="Save"/> <input type="button" value="Save as..."/> <input type="button" value="Format"/> or create a <input type="button" value="New query"/>

Promoters were calculated by the formula mentioned in the business knowledge template i.e. P-D.

It was seen that promoters on Facebook were 21.8% while those on Youtube were 2.6%.

This implies that referral marketing is more popular on Facebook, hence it should be targeted more.

- **Query 5:**

This query too is for the Campaign_Fact Table where we see the number of ads of each campaign running on each social media platform.

```
1 SELECT count(facebook.facebookid) AS facebookid,
2       count(youtube.youtubeid) AS youtubeid,
3       count(linkedin.linkedinid) AS linkednid,
4       count/twitter.twitterid) AS twitterid
5 FROM socialmedia
6 INNER JOIN advertising
7   ON socialmedia.socialmediaid = advertising.advertisingid
8 GROUP BY campaignid;
9
10
```

Cancel **Save** **Save as...** **Format** or create a **New query**

1	campaign id	facebookid	youtubeid	linkedinid	twitterid
2	1	25	8	14	2
3	2	34	10	3	2
4	3	35	0	22	0
5	5	20	12	3	5
6	6	0	0	20	30
7	7	0	30	0	20
8	8	0	0	21	29
9					

It can be seen that on average ads run on Facebook per campaign are more as compared to other social media platforms which also implies that most of the bank's audience comes from Facebook.