

# **Banking Dashboard**

## **Problem Statement –**

Develop a basic understanding of risk analytics in banking and financial services and understand how data is used to minimise the risk of losing money while lending to customers.

## **Solution –**

With our dashboards which are created using Power BI latest tools helps the company to make a decision based on the applicant's profile like if the applicant is likely to repay the loan then approving the loan otherwise not.

## **About Dataset –**


This dataset basically contains information about bank details ,various client details which consists of multiple tables which are interlinked with each other through keys like primary key and foreign key.

The various tables are Banking Relationship, Client-Banking, Gender, Investment Advisor and Period.

## **Data Cleaning –**

Creating a new column Engagement Timeframe in client-banking column which tells about the time line of the clients in banks.

Creating a new column Engagment Days in Client-Banking table how many days the client spent from the



Engagement days
3505
2602
8964
2690
1938
9300
2241
1924
2672
3719
1740
8294
1909
5510
1534
1433
7802
9123
1634
1421
2033
3890
6255
7081
7683

## Calculated Functions –

### Sum :

The power bi sum function will add all the numbers in a column and the column contains numbers to sum. It returns a decimal number.

### Syntax :

Sum= SUM(<column>)

Example:

Bank Deposit =

SUM('Clients – Banking'[Bank Deposits] )

**DistinctCount** :Counts the number of distinct values in a column

Syntax: DISTINCTCOUNT(<column>)

### Example :

Total Clients = DISTINCTCOUNT('Clients – Banking'[Client ID] )

**Sumx** :Returns the sum of an expression evaluated for each row in a table.

Syntax:

SUMX(<table>, <expression>)

Example :

Total Fees = SUMX('Clients – Banking' , [Total Loan] \* 'Clients – Banking'[Processing Fees] )

### **Switch :**

Evaluated an expression against a list of values and returns one of multiple possible result expressions

Syntax : SWITCH(<expression>, <value>, <result>[, <value>, <result>]...[, <else>])

### **DATEDIFF :**

Returns the number of interval boundaries between two dates.

Syntax :

DATEDIFF(<Date1>, <Date2>, <Interval>)

Example :

Engagment Days = DATEDIFF('Clients – Banking'[Joined Bank],TODAY(), DAY )

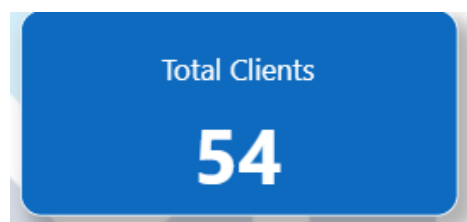
## KPI'S:

In which followings KPIS are present :

### Total Clients :

Total Clients KPI represents total number of clients in banking.

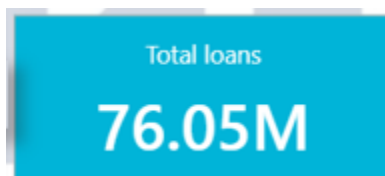
Total Clients =  $\text{DISTINCTCOUNT}(\text{'Clients – Banking' [Client ID]})$



### Total Loan :

Total Loan gives you information about the bank loan + Business lending + credit cards balance of particular investor , gender.

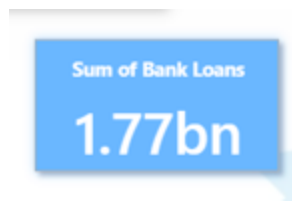
Total Loan =  $[\text{Bank Loan}] + [\text{Business Lending}] + [\text{Credit Cards Balance}]$



### Bank Loan :

Bank Loan gives you information what is the loan amount of loan to be repaid by the client to bank.

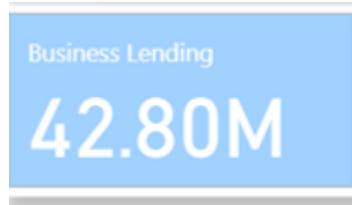
Bank Loan =  $\text{SUM}(\text{'Clients – Banking' [Bank Loans]})$



## Business Lending :

Business lending gives you information about the loan amount given to small business.

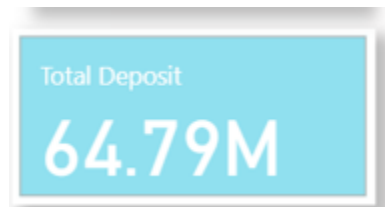
Business Lending = SUM('Clients – Banking'[Business Lending] )



## Total Deposit

Total Deposit gives you information about the amount deposited by particular investors in bank

Total Deposit = [Bank Deposit] + [Savings Account] + [Foreign Currency Account] + [Checking Accounts]



## Total Fees :

Total Fees is nothing but the amount charged by the bank for account set-up , maintenance charges etc.

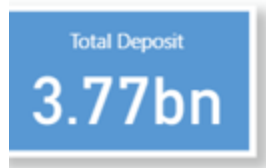
Total Fees = SUMX('Clients – Banking' , [Total Loan] \* 'Clients – Banking'[Processing Fees] )

## Bank Deposit :

Bank deposit is the money put in the bank.

Bank Deposit =

SUM('Clients – Banking'[Bank Deposits] )

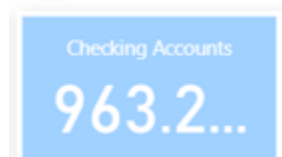


#### Checking Account Amount :

Checking account amount is nothing but which offers easy access to your money for daily transactional needs.

Checking Accounts =

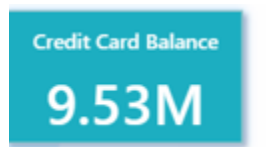
$\text{SUM}(\text{'Clients - Banking' [Checking Accounts]})$



#### Total CC Amount :

Total CC Amount is a short-term source of financing for a company by a bank.

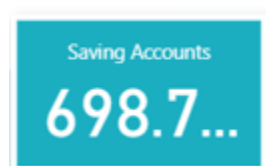
Total CC Amount =  $\text{SUM}(\text{'Clients - Banking' [Amount of Credit Cards]})$



#### Saving Account Amount :

A savings account is an interest-bearing deposit account held at a bank.

Savings Account =  $\text{SUM}(\text{'Clients - Banking' [Saving Accounts]})$



#### Engagement Account :

Engagement Banking is nothing but puts the customer at the center and aims to deliver the digital experiences they expect.

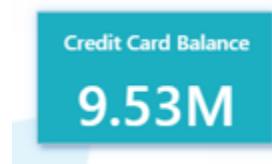
Engagment Length =

$SUM('Clients - Banking'[Engagment Days])$

**Credit Cards Balance :**

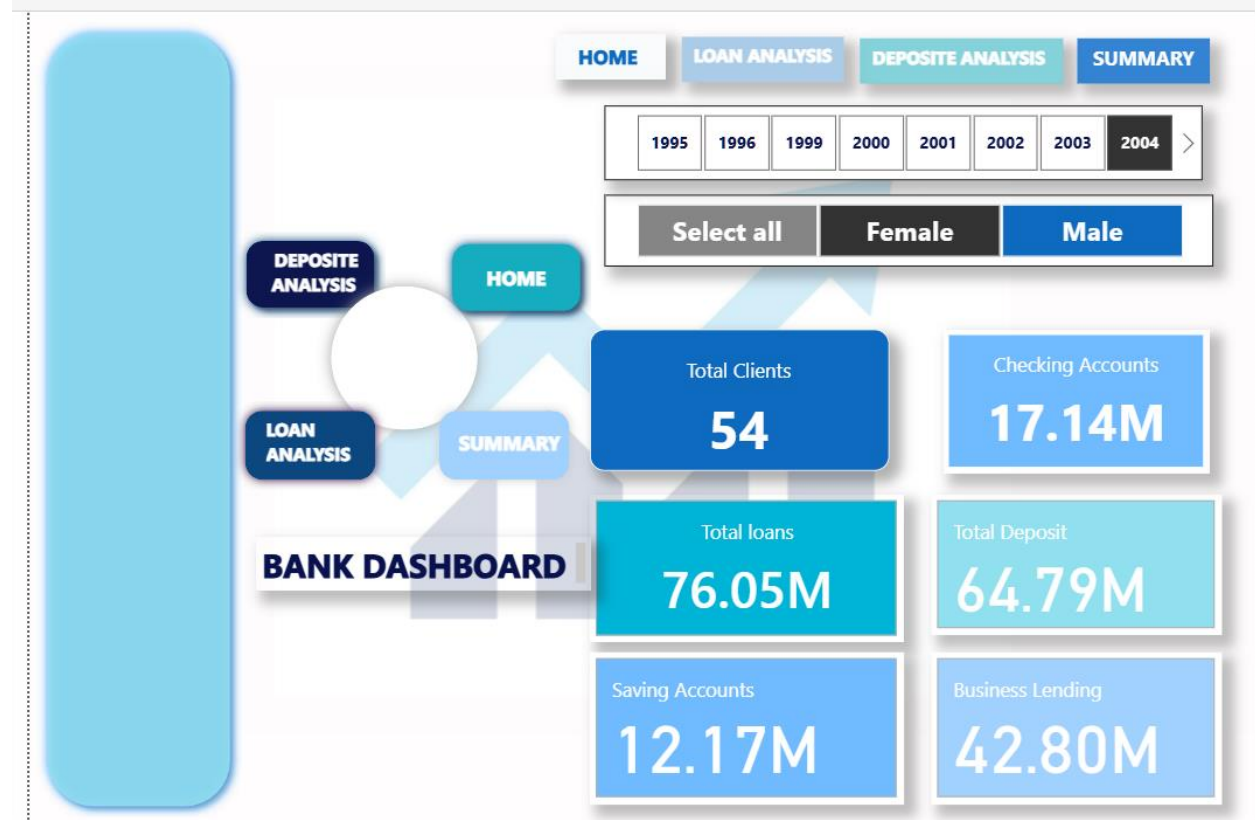
It is the total amount of money currently owned by a cardholder to their credit card bank.

Credit Cards Balance =  $SUM('Clients - Banking'[Credit Card Balance])$



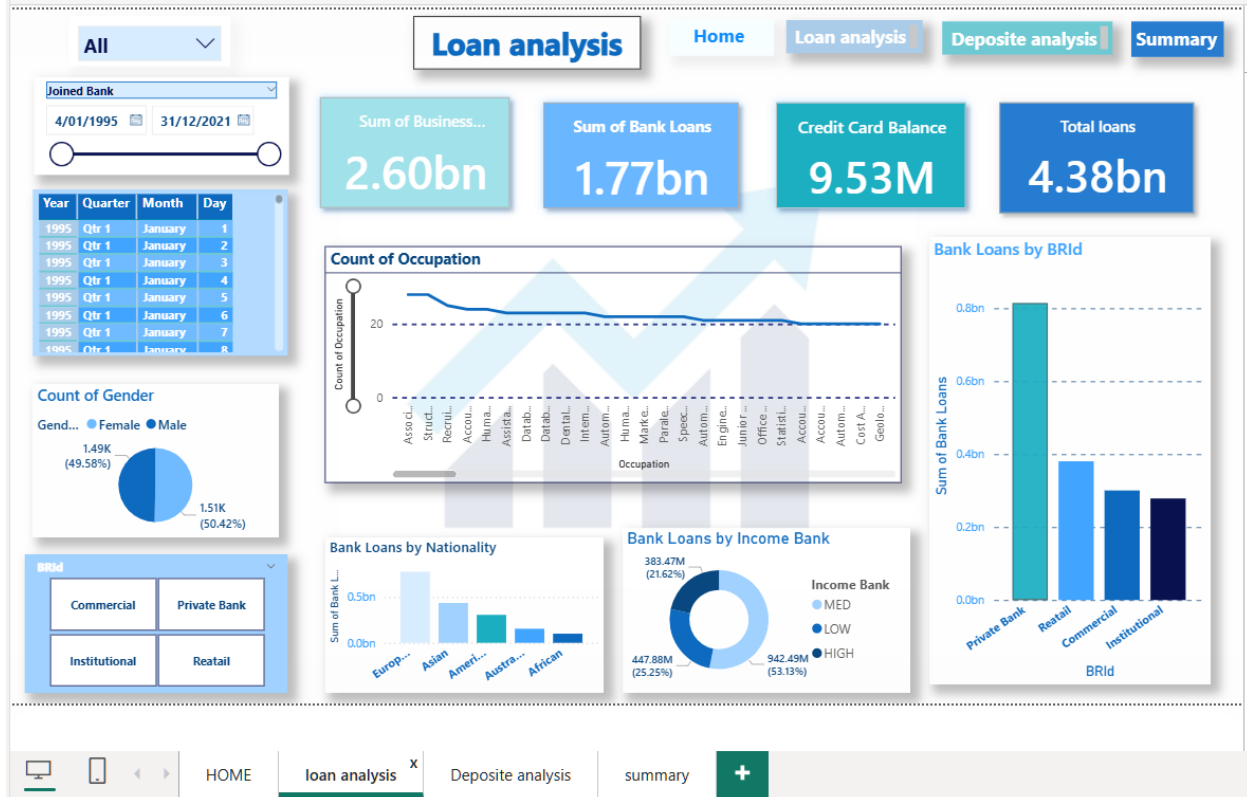
## Visualization And Result –

### Home

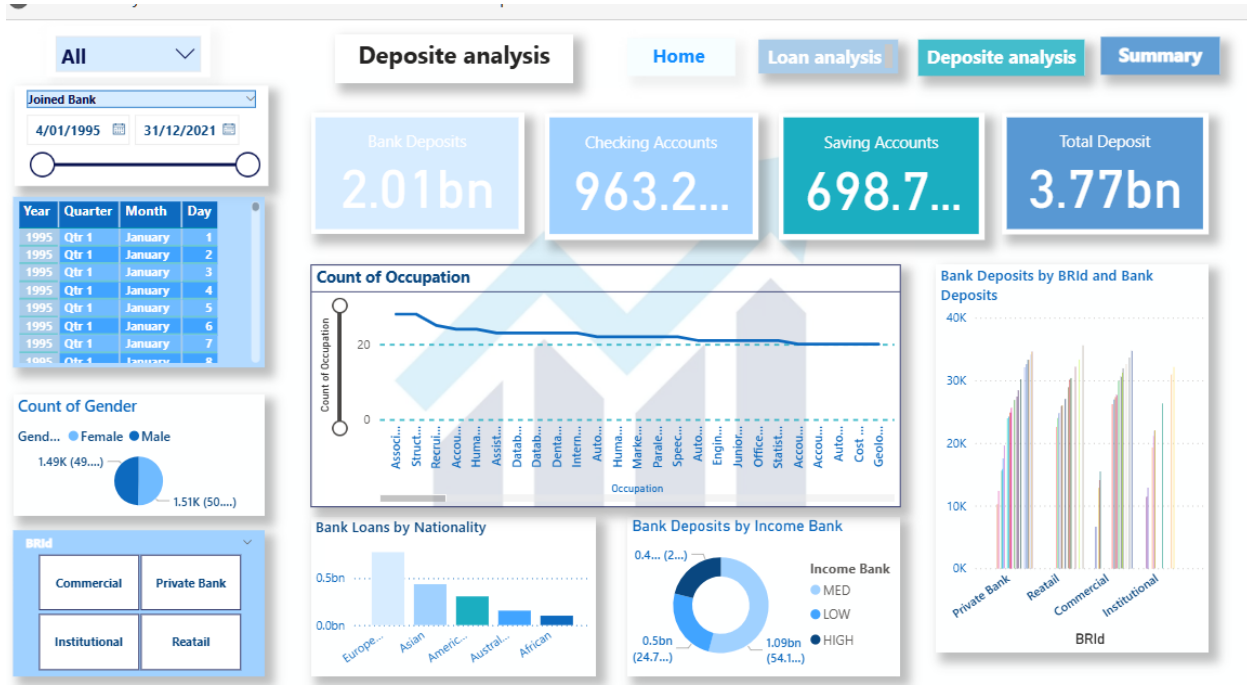


### Loan Analysis

Auto recovery contains some recovered files that haven't been opened.



## Deposit Analysis

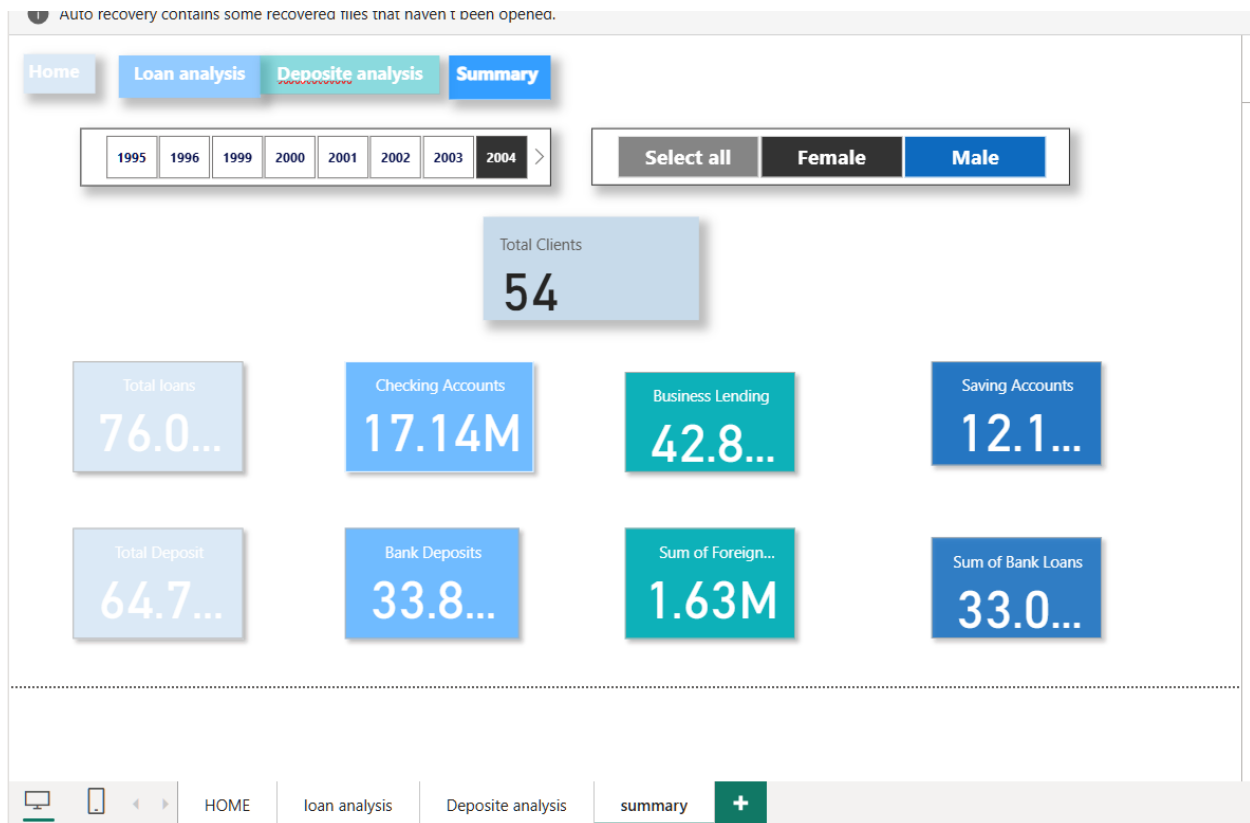




## Summary Dashboard

### Conclusion –

Empowered by the latest data visualization techniques, Power BI dashboards are among the most effective resources for using in banking sector. As outlined in this write-up, a banking operations dashboard in Power BI can be developed with key banking related metrics and KPIs.



### Future Work –

With these dashboards banks can easily know what is the total loan amount and all other things of a particular investor.

It also helps which type of banks have more number of clients as we can see private banks have more number of clients so it can help other banks can build their strategies to increase clients.

It also provides insights about which nationality has highest bank loans.

It gives information about various types of amount involved in different types of accounts by investors.

# MYSQL QUERY

## TOTAL TABLE :

7  
8  
9 `SELECT * FROM banking_case LIMIT 10;`

Client ID	Name	Age	Location ID	Joined Bank	Banking Contact	Nationality	Occupation	Fee Structure	Loyalty Class
IND81288	Raymond Mills	24	34324	06-05-2019	Anthony Torres	American	Safety Technician IV	High	Jade
IND65833	Julia Spencer	23	42205	10-12-2001	Jonathan Hawkins	African	Software Consultant	High	Jade
IND47499	Stephen Murray	27	7314	25-01-2010	Anthony Berry	European	Help Desk Operator	High	Gold
IND60181	Virginia Garza	40	34594	28-03-2019	Steve Diaz	American	Geologist II	Mid	Silver
IND78532	Melesa Sanders	46	41269	20-07-2012	Shawn Long	American	Assistant Professor	Mid	Platinum
IND95683	Samuel Hudson	23	13204	07-02-2019	Douglas Tucker	American	Help Desk Technician	High	Silver
IND40785	Timothy Alexander	46	42910	02-06-2002	Douglas Tucker	Asian	Account Coordinator	High	Gold
IND13570	Carl Martin	78	6127	03-11-2000	Steve Diaz	European	Automation Specialist II	Mid	Gold
IND53299	Philip Day	67	32656	07-04-2015	Bruce Butler	Asian	Software Test Engineer II	High	Silver
	Jason Sims	51	28340	20-11-1995	Joe Price	European	Geologist III	Mid	Silver

11 • `DESCRIBE banking_case;`

12

Field	Type	Null	Key	Default	Extra
Client ID	text	YES		NULL	
Name	text	YES		NULL	
Age	int	YES		NULL	
Location ID	int	YES		NULL	
Joined Bank	text	YES		NULL	
Banking Contact	text	YES		NULL	
Nationality	text	YES		NULL	
Occupation	text	YES		NULL	
Fee Structure	text	YES		NULL	
Loyalty Classification	text	YES		NULL	
Estimated Income	dou...	YES		NULL	
Superannuation Sa...	dou...	YES		NULL	
Amount of Credit C...	int	YES		NULL	
Credit Card Balance	dou...	YES		NULL	
Bank Loans	dou...	YES		NULL	

Result 4 customer 5 Result 6 x

## Total deposit

```

26      COUNT(DISTINCT client_id) AS total_clients
27 FROM banking_case;
28
29 • SELECT
30     SUM(`Bank Deposits`) AS total_deposits
31 FROM banking_case;
32

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
total_deposits			
▶ 2014680581.770001			

### Total loan

```

31 FROM banking_case;
32
33 • SELECT
34     SUM(`Bank Loans`) AS total_loans
35 FROM banking_case;
36

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
total_loans			
▶ 1774158466.4599965			

### Avg credit card balance

```

35 FROM banking_case;
36 • SELECT
37     AVG(`Credit Card Balance`) AS avg_credit_balance
38 FROM banking_case;
39

```

avg_credit_balance
3176.706043333333

### Average Account Balance

```

38 FROM banking_case;
39
40 • SELECT
41     AVG(`Checking Accounts` + `Saving Accounts` + `Foreign Currency Account`) AS avg_total_balance
42 FROM banking_case;
43

```

avg_total_balance
583884.8326033333

### Active vs Inactive Clients

```

42 FROM banking_case;
43
44 • SELECT
45     CASE
46         WHEN `Bank Deposits` > 0 OR `Bank Loans` > 0 THEN 'Active'
47         ELSE 'Inactive'
48     END AS client_status,
49     COUNT(*) AS total_clients
50 FROM banking_case
51 GROUP BY client_status;
52

```

client_status	total_clients
Active	3000

### Engagement Duration Buckets

```

51 GROUP BY client_status;
52
53 SELECT
54 CASE
55 WHEN DATEDIFF(CURDATE(), STR_TO_DATE('Joined Bank', '%d-%m-%Y')) < 365 THEN '< 1 Year'
56 WHEN DATEDIFF(CURDATE(), STR_TO_DATE('Joined Bank', '%d-%m-%Y')) < 1825 THEN '< 5 Years'
57 WHEN DATEDIFF(CURDATE(), STR_TO_DATE('Joined Bank', '%d-%m-%Y')) < 3650 THEN '< 10 Years'
58 ELSE '> 10 Years'
59 END AS engagement_duration,
60 COUNT(*) AS client_count
61 FROM banking_case
62 GROUP BY engagement_duration;
63

```

engagement_duration	client_count
< 10 Years	694
> 10 Years	2010
< 5 Years	296

## Top 5 Branches by Total Deposits

```

70 LIMIT 5;
71
72 SELECT
73 BRId,
74 ROUND(SUM(`Bank Loans`) / NULLIF(SUM(`Bank Deposits`), 0), 2) AS loan_to_deposit_ratio
75 FROM banking_case
76 GROUP BY BRId;
77

```

BRId	loan_to_deposit_ratio
1	0.9
2	0.85
3	0.88
4	0.9

## Loan-to-Deposit Ratio by Branch

```

62     GROUP BY engagement_duration;
63
64 • SELECT
65     BRId,
66     SUM(`Bank Deposits`) AS total_deposits
67 FROM banking_case
68 GROUP BY BRId
69 ORDER BY total_deposits DESC
70 LIMIT 5;
71

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
BRId	total_deposits			
3	925398533.850001			
1	425492375.85999995			
4	334821728.42			
2	328967943.64000005			

L

### Client Risk Profile Distribution

```

76     GROUP BY BRId;
77
78 • SELECT
79     `Risk Weighting`,
80     COUNT(*) AS total_clients
81 FROM banking_case
82 GROUP BY `Risk Weighting`
83 ORDER BY `Risk Weighting`;
84

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	Risk Weighting	total_clients			
▶	1	836			
	2	1222			
	3	460			
	4	322			
	5	160			

### Total Bank Deposits, Loans, and Business Lending

```

76 GROUP BY BRId;
77
78 • SELECT
79     `Risk Weighting`,
80     COUNT(*) AS total_clients
81 FROM banking_case
82 GROUP BY `Risk Weighting`
83 ORDER BY `Risk Weighting`;
84
85 • SELECT
86     SUM(`Bank Deposits`) AS total_deposits,
87     SUM(`Bank Loans`) AS total_loans,
88     SUM(`Business Lending`) AS total_business_lending
89 FROM banking_case;
90

```

total_deposits	total_loans	total_business_lending
2014680581.770001	1774158466.4599965	2600279425.2200036

### Average Balance Per Account Type

```

89 FROM banking_case;
90
91 • SELECT
92     AVG(`Checking Accounts`) AS avg_checking,
93     AVG(`Saving Accounts`) AS avg_saving,
94     AVG(`Foreign Currency Account`) AS avg_foreign_currency
95 FROM banking_case;
96

```

avg_checking	avg_saving	avg_foreign_currency
321092.94912666606	232908.3534833335	29883.52999333338

### Bank Clients by Loyalty Classification

```

95 FROM banking_case;
96
97 • SELECT
98     `Loyalty Classification`,
99     COUNT(*) AS total_clients
100 FROM banking_case
101 GROUP BY `Loyalty Classification`
102 ORDER BY total_clients DESC;
103

```

Result Grid		Filter Rows:	Export:	Wrap Cell Contents:
	Loyalty Classification	total_clients		
▶	Jade	1331		
	Silver	767		
	Gold	585		
	Platinum	317		

### Branch-wise Total Deposits

```

102 ORDER BY total_clients DESC;
103
104 • SELECT
105     BRId,
106     SUM(`Bank Deposits`) AS branch_total_deposits
107 FROM banking_case
108 GROUP BY BRId
109 ORDER BY branch_total_deposits DESC;
110

```

Result Grid		Filter Rows:	Export:	Wrap Cell Contents:
	BRId	branch_total_deposits		
▶	3	925398533.850001		
	1	425492375.85999995		
	4	334821728.42		
	2	328967943.64000005		

### Average Credit Card Usage



```
109 ORDER BY branch_total_deposits DESC;
110
111 • SELECT
112     AVG(`Credit Card Balance`) AS avg_credit_card_balance,
113     AVG(`Amount of Credit Cards`) AS avg_credit_card_count
114 FROM banking_case;
115
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	avg_credit_card_balance	avg_credit_card_count
▶	3176.206943333331	1.4637