**TERADATA Certification Project Report -Edureka**

Manoj K V

05/13/2017

**Problem Statement**

The project should be able to segment the banking customers according to the business rules defined. Customer segmentation is the process of dividing the customers into groups who have similar characteristics in terms of their balance in various products.

**Business Rules:**

1. The project should segment the customers who hold Saving Accounts, Credit Card accounts and Mortgage accounts.

2. The project should exclude other accounts.

3. Customers should be segmented as following categories.

Low-Networth, Medium-Networth or High-Networth customers (Based on their balance held in the accounts)

4. The project should build a target table that contains customers and their corresponding segments.

**Criteria for Segmentation:**

1. If the customer meets *one of the below criteria*, then the segment of the customer will be

***High-Networth.***

Saving accounts balance > 5, 00,000

Credit Card Balance > 3, 00,000

Mortgage account amount > 50, 00,000

2. If the customer meets *one of the below criteria*, then the segment of the customer will be

***Medium-Networth***

Saving accounts balance > 2, 00,000

Credit Card Balance > 1, 00,000

Mortgage account amount > 10, 00,000

3. If the customer meets *one of the below criteria*, then the segment of the customer will be

***Low-Networth***

Saving accounts balance < 2, 00,000

Credit Card Balance < 1, 00,000

Mortgage account amount < 10, 00,000

**Source system**

Source system would provide below data:

* Customer - All customers of the bank
* Accounts - All open accounts
* Account Type – Reference table for accounts like Savings, Credit Card, Mortgage, and PPF.

**DDLs:**

Created the staging and target tables as per structure provided.

*--Customer\_Stg;*

**CREATE** **MULTISET** **TABLE** edureka.Customer\_Stg ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_ID INTEGER,

Customer\_FName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_LName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Dob DATE FORMAT 'yyyy-mm-dd',

Customer\_City CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_State CHAR(2) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Phone CHAR(10) CHARACTER **SET** LATIN **NOT** CASESPECIFIC)

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_ID )**;**

*--Customer;*

**CREATE** **SET** **TABLE** edureka.Customer ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_ID INTEGER,

Customer\_FName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_LName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Dob DATE FORMAT 'yyyy-mm-dd',

Customer\_City CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_State CHAR(2) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Phone CHAR(10) CHARACTER **SET** LATIN **NOT** CASESPECIFIC)

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_ID )**;**

*--Accounts\_Stg;*

**CREATE** **MULTISET** **TABLE** edureka.Accounts\_Stg ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_Id INTEGER,

Account\_ID INTEGER,

Account\_Open\_Dt DATE FORMAT 'yyyy-mm-dd',

Account\_Close\_Dt DATE FORMAT 'yyyy-mm-dd',

Account\_Type\_Cd BYTEINT,

Account\_Balance INTEGER)

**UNIQUE** **PRIMARY** **INDEX** ( Account\_ID )**;**

*--Accounts;*

**CREATE** **SET** **TABLE** edureka.Accounts ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_Id INTEGER,

Account\_ID INTEGER,

Account\_Open\_Dt DATE FORMAT 'yyyy-mm-dd',

Account\_Close\_Dt DATE FORMAT 'yyyy-mm-dd',

Account\_Type\_Cd BYTEINT,

Account\_Balance INTEGER)

**UNIQUE** **PRIMARY** **INDEX** ( Account\_ID )

**INDEX** ( Customer\_Id )**;**

*--Account\_Type;*

**CREATE** **SET** **TABLE** edureka.Account\_Type ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Account\_Type\_Cd BYTEINT,

Account\_Type\_Desc CHAR(10) CHARACTER **SET** LATIN **NOT** CASESPECIFIC)

**UNIQUE** **PRIMARY** **INDEX** ( Account\_Type\_Cd )**;**

*--Customer\_First\_Account;*

**CREATE** **SET** **TABLE** edureka.Customer\_First\_Account ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_Id INTEGER,

First\_Account\_Dt DATE FORMAT 'yyyy-mm-dd')

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_Id )**;**

*--Customer\_Accounts;*

**CREATE** **SET** **TABLE** edureka.Customer\_Accounts ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_Id INTEGER,

Savings\_Balance INTEGER,

Credit\_Balance INTEGER,

Mortgage\_Balance INTEGER)

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_Id )**;**

*--Customer\_Segment;*

**CREATE** **SET** **TABLE** edureka.Customer\_Segment ,**NO** **FALLBACK** ,

**NO** **BEFORE** **JOURNAL**,

**NO** **AFTER** **JOURNAL**,

**CHECKSUM** = **DEFAULT**,

**DEFAULT** **MERGEBLOCKRATIO**

(

Customer\_Id INTEGER,

Customer\_FName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_LName CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Type CHAR(1) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Customer\_Segment\_Value CHAR(15) CHARACTER **SET** LATIN **NOT** CASESPECIFIC,

Segment\_Start\_Dt DATE FORMAT 'yyyy-mm-dd',

Segment\_End\_Dt DATE FORMAT 'yyyy-mm-dd')

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_Id )**;**

**Scripts:**

1. Written a fastload script named as ***customerload.fl***to load the given ***Customer.txt*** into staging table ***Customer\_Stg***.

fastload <<EOF

sessions 16**;**

**logon** 127.0.0.1/dbc,dbc**;**

**DROP** **TABLE** edureka.UV\_Customer\_Stg**;**

**DROP** **TABLE** edureka.ET\_Customer\_Stg**;**

**delete** **from** edureka.Customer\_Stg**;**

**set** record vartext ","**;**

define

Customer\_ID (varchar(15)),

Customer\_FName (varchar(15)),

Customer\_LName (varchar(15)),

Customer\_Dob (varchar(15)),

Customer\_City (varchar(15)),

Customer\_State (varchar(2)),

Customer\_Phone (varchar(10)),

**file**=Customer.txt

**;**

**begin** **loading** edureka.Customer\_Stg

**errorfiles** edureka.ET\_Customer\_Stg, edureka.UV\_Customer\_Stg

**;**

**insert** **into** edureka.Customer\_Stg

(

:Customer\_ID,

:Customer\_FName,

:Customer\_LName,

:Customer\_Dob,

:Customer\_City,

:Customer\_State,

:Customer\_Phone

)**;**

**end** **loading;**

logoff**;**

EOF

2. Written a fastload script named as ***accountsload.fl*** to load the below ***Accounts.txt*** into staging table ***Accounts\_Stg***;

fastload <<EOF

sessions 16**;**

**logon** 127.0.0.1/dbc,dbc**;**

**DROP** **TABLE** edureka.UV\_Accounts\_Stg**;**

**DROP** **TABLE** edureka.ET\_Accounts\_Stg**;**

**delete** **from** edureka.Accounts\_Stg**;**

**set** record vartext ","**;**

define

Customer\_ID (varchar(15)),

Account\_ID (varchar(15)),

Account\_Open\_Dt (varchar(15)),

Account\_Close\_Dt (varchar(15)),

Account\_Type\_Cd (varchar(15)),

Account\_Balance (varchar(15)),

**file**=Accounts.txt

**;**

**begin** **loading** edureka.Accounts\_Stg

**errorfiles** edureka.ET\_Accounts\_Stg, edureka.UV\_Accounts\_Stg

**;**

**insert** **into** edureka.Accounts\_Stg

(

:Customer\_Id,

:Account\_ID,

:Account\_Open\_Dt,

:Account\_Close\_Dt,

:Account\_Type\_Cd,

:Account\_Balance

)**;**

**end** **loading;**

logoff**;**

EOF

3. Written insert queries to load the given records into ***Account\_type*** table

**insert** **into** edureka.Account\_type (Account\_Type\_Cd,Account\_Type\_Desc ) **values** (01,'Savings')**;**

**insert** **into** edureka.Account\_type (Account\_Type\_Cd,Account\_Type\_Desc ) **values**

(02,'Credit Card')**;**

**insert** **into** edureka.Account\_type (Account\_Type\_Cd,Account\_Type\_Desc ) **values** (03,'Mortgage')**;**

**insert** **into** edureka.Account\_type (Account\_Type\_Cd,Account\_Type\_Desc ) **values**

(04,'PPF')**;**

**select** \* **from** edureka.Account\_type**;**

*/\**

*Account\_Type\_Cd Account\_Type\_Desc*

*1 Savings*

*2 Credit Car*

*3 Mortgage*

*4 PPF*

*\*/*

4.Written a BTEQ script named as ***CustomerAccountsLoad.bteq*** to load the ***Customer*** and ***Accounts*** table from their corresponding staging tables ***Customer\_Stg*** and ***Accounts\_Stg***; (BTEQ script also have the delete queries to delete the existing records from ***Customer*** and ***Accounts*** tables and load from ***Customer\_Stg*** and ***Accounts\_Stg***);

bteq <<EOF

.**LOGON** 127.0.0.1/dbc,dbc**;**

.MAXERROR 1

*--Customer Table*

**DELETE** **FROM** edureka.Customer**;**

**INSERT** **INTO** edureka.Customer

(

Customer\_ID,

Customer\_FName,

Customer\_LName,

Customer\_Dob,

Customer\_City,

Customer\_State,

Customer\_Phone

)

**select**

STG.Customer\_ID,

STG.Customer\_FName,

STG.Customer\_LName,

STG.Customer\_Dob,

STG.Customer\_City,

STG.Customer\_State,

STG.Customer\_Phone

**FROM** edureka.Customer\_Stg STG

**;**

*--Accounts Table*

**DELETE** **FROM** edureka.Accounts **;**

**INSERT** **INTO** edureka.Accounts

(

Customer\_Id,

Account\_ID,

Account\_Open\_Dt,

Account\_Close\_Dt,

Account\_Type\_Cd,

Account\_Balance

)

**select**

STG.Customer\_Id,

STG.Account\_ID,

STG.Account\_Open\_Dt,

STG.Account\_Close\_Dt,

STG.Account\_Type\_Cd,

STG.Account\_Balance

**FROM** edureka.Accounts\_Stg STG

**;**

.LOGOFF

**.QUIT** 0**;**

EOF

5. Written a BTEQ script named as ***CustomerSegment.bteq***. This BTEQ script has below steps.

bteq <<EOF

.**LOGON** 127.0.0.1/dbc,dbc**;**

.MAXERROR 1

*/\*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*

*Write a BTEQ script named as CustomerSegment.bteq. This BTEQ script should have below steps.*

*/\*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*

*/\*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*

*1.Create a Volatile table named as "Customer\_First\_Account" with columns Customer\_Id and First\_Account\_Dt.*

*Write an Insert query to identify the first account open date for each customer from Accounts table and*

*insert into "Customer\_First\_Account" table.*

*If the customer has multiple accounts, then the oldest Account\_Open\_Dt should be considered.*

*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\*/*

**CREATE** **MULTISET** **VOLATILE** **TABLE** Customer\_First\_Account

(

Customer\_Id INTEGER,

First\_Account\_Dt DATE FORMAT 'yyyy-mm-dd'

)

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_Id )

**ON** **COMMIT** **PRESERVE** **ROWS;**

**INSERT** **INTO** Customer\_First\_Account

(

Customer\_Id,

First\_Account\_Dt

)

**SELECT**

accnt.Customer\_Id,

accnt.Account\_Open\_Dt **as** First\_Account\_Dt

**FROM** edureka.Accounts accnt

**QUALIFY** ROW\_NUMBER () OVER (**partition** **by** Customer\_Id **order** **by** Account\_Open\_Dt)=1 *--first account open date/customer*

**;**

*/\*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*

*2.Create a Volatile table named as "Customer\_Accounts" with columns Customer\_Id, Savings\_Balance, Credit\_Balance and Mortgage\_Balance.*

*Write an insert query to calculate the savings account balance, credit card balance and mortgage balance*

*from Accounts table for each customer and insert into "Customer\_Accounts" table.*

*For each customer, there should be only one record in this table which contains their savings account balance, credit card balance and mortgage balance.*

*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\*/*

**CREATE** **MULTISET** **VOLATILE** **TABLE** Customer\_Accounts

(

Customer\_Id INTEGER,

Savings\_Balance INTEGER,

Credit\_Balance INTEGER,

Mortgage\_Balance INTEGER)

**UNIQUE** **PRIMARY** **INDEX** ( Customer\_Id )

**ON** **COMMIT** **PRESERVE** **ROWS;**

**INSERT** **INTO** Customer\_Accounts

(

Customer\_Id,

Savings\_Balance,

Credit\_Balance,

Mortgage\_Balance

)

**SELECT**

bal.Customer\_Id,

bal.Savings\_Balance,

bal.Credit\_Balance,

bal.Mortgage\_Balance

**FROM**

(

**SELECT**

Customer\_Id,

sum(coalesce( **case** **when** Account\_Type\_Cd=1 **then** Account\_Balance **end**, 0)) **as** Savings\_Balance,

sum(coalesce( **case** **when** Account\_Type\_Cd=2 **then** Account\_Balance **end**, 0)) **as** Credit\_Balance,

sum(coalesce( **case** **when** Account\_Type\_Cd=3 **then** Account\_Balance **end**, 0)) **as** Mortgage\_Balance,

sum(coalesce( **case** **when** Account\_Type\_Cd=4 **then** Account\_Balance **end**, 0)) **as** PPF\_Balance

**FROM** edureka.Accounts accnt

**WHERE** Account\_Type\_Cd <> 4 *--Excluding PPF*

**GROUP** **BY** 1

) bal

**;**

*/\*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*

*3.Write a delete query to delete existing data from "customer\_segment" table*

*Insert into "Customer\_Segment" table from Customer, "Customer\_Accounts" and "Customer\_First\_Account" tables. Criteria for each column is given below.*

*Column Customer\_Type is set to N, if the first account opened date (First\_Account\_Dt from "Customer\_First\_Account" table) of the customer is within last 3 months, otherwise it is set to E.*

*Column "Customer\_Segment"ation\_Value is calculated based on the segmentation criteria provided in problem statement.*

*Compare the Savings\_Balance, Credit\_Balance and Mortgage\_Balance against the criteria provided and assign "Customer\_Segment"ation\_value as High\_Networth, Medium\_Networth or Low\_Networth.*

*Column Segmentation\_start\_dt should be set to current date and segmentation\_end\_dt should be set to 9999-12-31.*

*@@@@@@@@@@@@@@@@@@@@@@@@@@22@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\*/*

**DELETE** **FROM** edureka.customer\_segment**;**

**INSERT** **INTO** edureka.customer\_segment

(

Customer\_Id,

Customer\_FName,

Customer\_LName,

Customer\_Type,

Customer\_Segment\_Value,

Segment\_Start\_Dt,

Segment\_End\_Dt

)

**SELECT**

cust.Customer\_Id,

cust.Customer\_FName,

cust.Customer\_LName,

**case**

**when** cfa.First\_Account\_Dt between (CURRENT\_DATE- INTERVAL '3' MONTH) **and** (CURRENT\_DATE) **then** 'N'

**else** 'E'

**end** **as** Customer\_Type,

**CASE**

**WHEN** ca.Savings\_Balance > 500000 **OR** ca.Credit\_Balance > 300000 **OR** ca.Mortgage\_Balance > 5000000 **THEN** 'High-Networth'

**WHEN** ca.Savings\_Balance > 200000 **OR** ca.Credit\_Balance > 100000 **OR** ca.Mortgage\_Balance > 1000000 **THEN** 'Medium-Networth'

**WHEN** ca.Savings\_Balance < 200000 **OR** ca.Credit\_Balance < 100000 **OR** ca.Mortgage\_Balance < 1000000 **THEN** 'Low-Networth'

**ELSE** 'Others'

**END** **as** Customer\_Segment\_Value,

CURRENT\_DATE **as** Segment\_Start\_Dt,

cast('9999-12-31' **as** date) **as** Segment\_End\_Dt

**FROM** edureka.Customer cust

**JOIN** Customer\_Accounts ca

**ON** cust.Customer\_Id = ca.Customer\_Id

**JOIN** Customer\_First\_Account cfa

**ON** cust.Customer\_Id = cfa.Customer\_Id

**;**

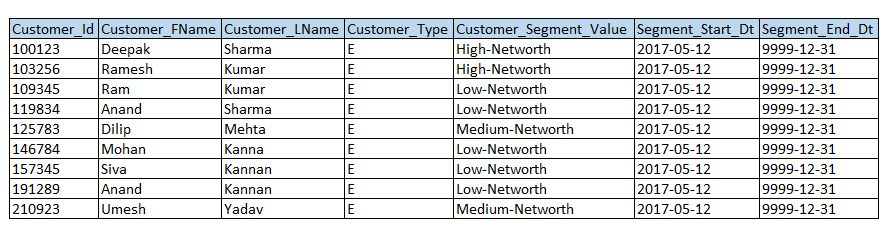
.LOGOFF

**.QUIT** 0**;**

EOF

**Output**

**SELECT** \* **FROM** edureka.customer\_segment **ORDER** **BY** Customer\_Id**;**



**Appendix**



**Thank you!**