SQL-Mongo Project - Home Credit Data

BUAN 6320.001

Group Members

Pavan Sai Krishna Gorantla

Aashesh Nareshchand

Shashank Srivastava

Colby Porter

Group #: 5

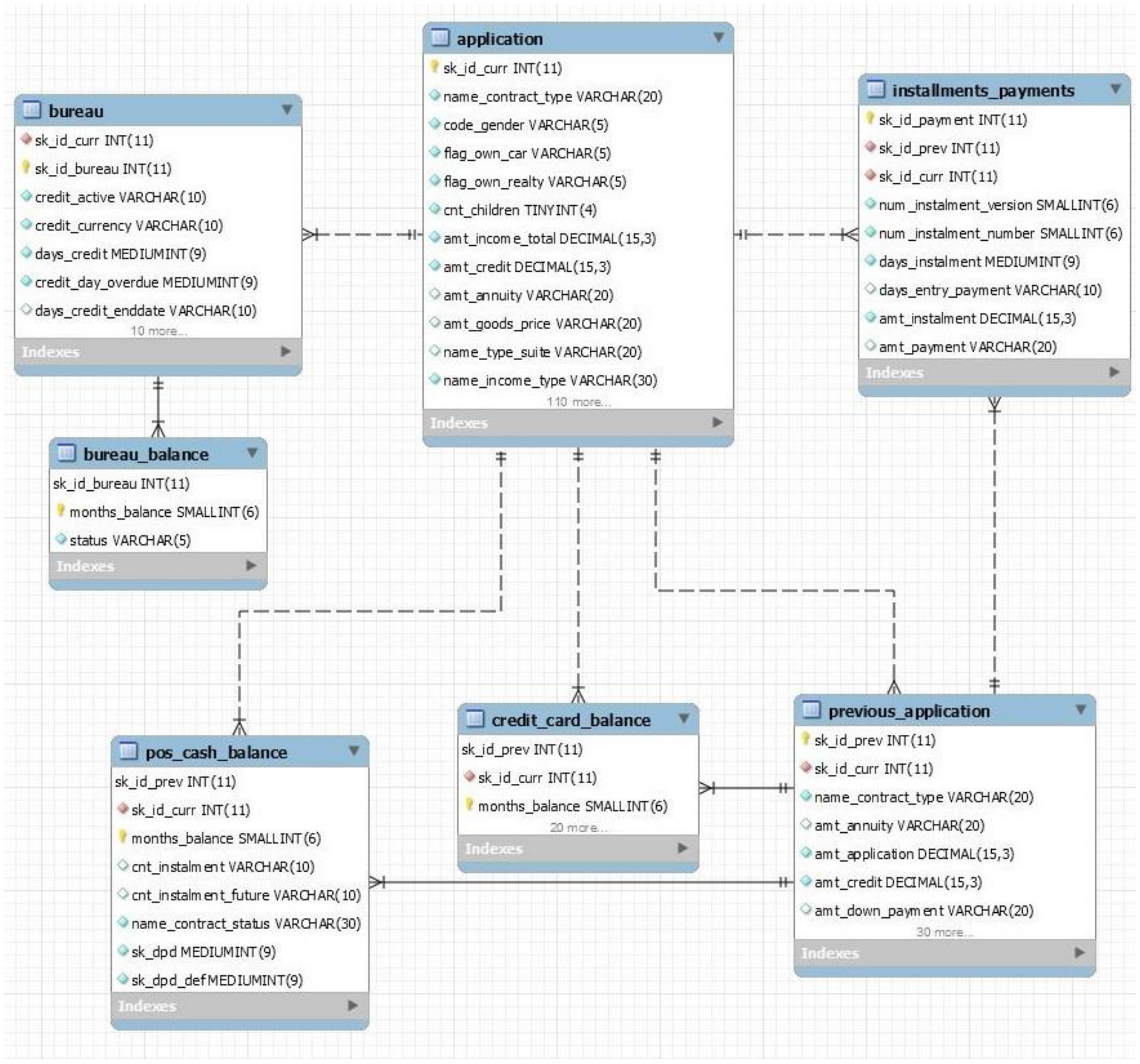
Contents

Data Model

Assumptions/Notes About Data Entities and Relationships

application - each row represents one loan application and is identified by sk_id_curr as the primary key
previous_application - each row is an old application, identified by sk_id_prev, associated to sk_id_curr
bureau - each row is credit info with credit bureau, identified by sk_id_bureau, associated to sk_id_curr
bureau_balance - each row is one month balance of previous credits in association with sk_id_bureau
pos_cash_balance - each row is one month balance of point of sale loans in association with sk_id_prev
credit_card_balance - each row is one month balance of credit card loans in association with sk_id_prev
installments_payments - each row records every payment done or missed, identified by sk_id_payment

Entity-Relationship Diagram



Physical Database

Assumptions/Notes About Data Set

Merged application test and application train datasets with TARGET column as blank for test dataset
Many columns in almost all the tables have missing values and are uploaded into database as blanks

Screen shot of Physical Database objects (tables on left side and indexes on right side)

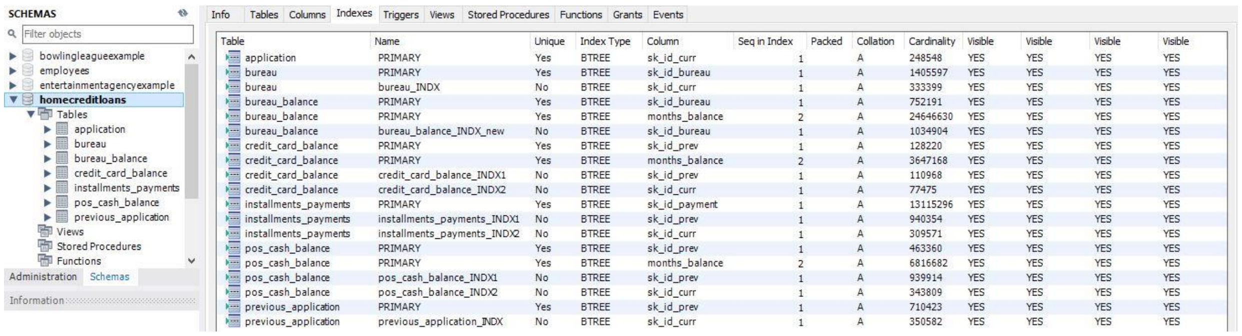
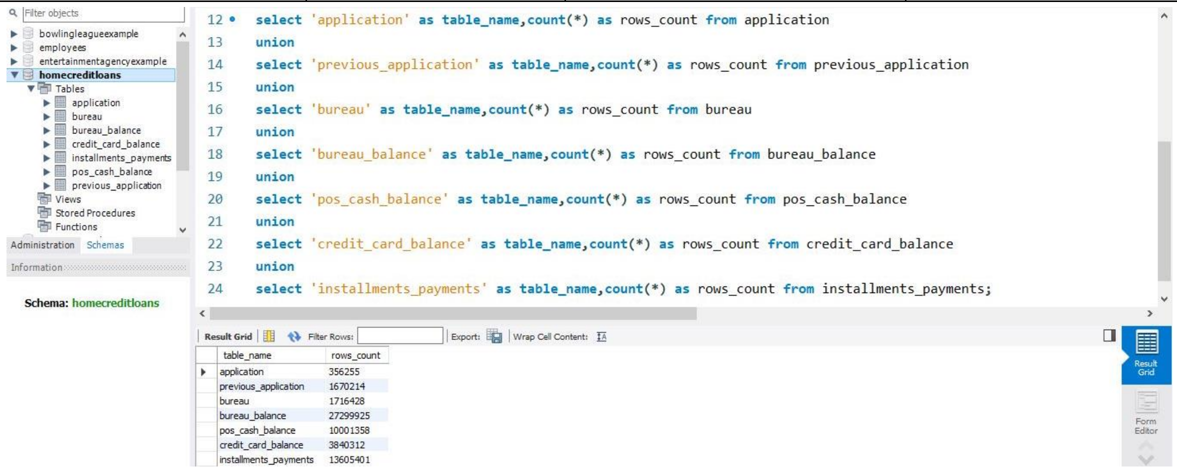


Table	Name	Unique	Index Type	Column	Seq in Index	Packed	Collation	Cardinality	Visible	Visible	Visible	Visible
application	PRIMARY	Yes	BTREE	sk_id_curr	1		A	248548	YES	YES	YES	YES
bureau	PRIMARY	Yes	BTREE	sk_id_bureau	1		A	1405597	YES	YES	YES	YES
bureau	bureau_INDX	No	BTREE	sk_id_curr	1		A	333399	YES	YES	YES	YES
bureau_balance	PRIMARY	Yes	BTREE	sk_id_bureau	1		A	752191	YES	YES	YES	YES
bureau_balance	PRIMARY	Yes	BTREE	months_balance	2		A	24646630	YES	YES	YES	YES
bureau_balance	bureau_balance_INDX_new	No	BTREE	sk_id_bureau	1		A	1034904	YES	YES	YES	YES
credit_card_balance	PRIMARY	Yes	BTREE	sk_id_prev	1		A	128220	YES	YES	YES	YES
credit_card_balance	PRIMARY	Yes	BTREE	months_balance	2		A	3647168	YES	YES	YES	YES
credit_card_balance	credit_card_balance_INDX1	No	BTREE	sk_id_prev	1		A	110968	YES	YES	YES	YES
credit_card_balance	credit_card_balance_INDX2	No	BTREE	sk_id_curr	1		A	77475	YES	YES	YES	YES
installments_payments	PRIMARY	Yes	BTREE	sk_id_payment	1		A	13115296	YES	YES	YES	YES
installments_payments	installments_payments_INDX1	No	BTREE	sk_id_prev	1		A	940354	YES	YES	YES	YES
installments_payments	installments_payments_INDX2	No	BTREE	sk_id_curr	1		A	309571	YES	YES	YES	YES
pos_cash_balance	PRIMARY	Yes	BTREE	sk_id_prev	1		A	463360	YES	YES	YES	YES
pos_cash_balance	PRIMARY	Yes	BTREE	months_balance	2		A	6816682	YES	YES	YES	YES
pos_cash_balance	pos_cash_balance_INDX1	No	BTREE	sk_id_prev	1		A	939914	YES	YES	YES	YES
pos_cash_balance	pos_cash_balance_INDX2	No	BTREE	sk_id_curr	1		A	343809	YES	YES	YES	YES
previous_application	PRIMARY	Yes	BTREE	sk_id_prev	1		A	710423	YES	YES	YES	YES
previous_application	previous_application_INDX	No	BTREE	sk_id_curr	1		A	350582	YES	YES	YES	YES

Data in the Database

Table Name	Primary Key	Foreign Key	# of Rows in Table
application	sk_id_curr	--	356,255
previous_application	sk_id_prev	sk_id_curr	1,670,214
bureau	sk_id_bureau	sk_id_curr	1,716,428
bureau_balance	sk_id_bureau, months_balance	sk_id_bureau	27,299,925
pos_cash_balance	sk_id_prev, months_balance	sk_id_curr, sk_id_prev	10,001,358
credit_card_balance	sk_id_prev, months_balance	sk_id_curr, sk_id_prev	3,840,312
installments_payments	sk_id_payment*	sk_id_curr, sk_id_prev	13,605,401



```
select 'application' as table_name,count(*) as rows_count from application
union
select 'previous_application' as table_name,count(*) as rows_count from previous_application
union
select 'bureau' as table_name,count(*) as rows_count from bureau
union
select 'bureau_balance' as table_name,count(*) as rows_count from bureau_balance
union
select 'pos_cash_balance' as table_name,count(*) as rows_count from pos_cash_balance
union
select 'credit_card_balance' as table_name,count(*) as rows_count from credit_card_balance
union
select 'installments_payments' as table_name,count(*) as rows_count from installments_payments;
```

table_name	rows_count
application	356255
previous_application	1670214
bureau	1716428
bureau_balance	27299925
pos_cash_balance	10001358
credit_card_balance	3840312
installments_payments	13605401

SQL Queries

Query 1

Question

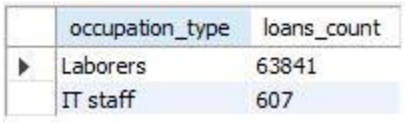
Which occupation types had the highest and lowest number of loans?

Translation

Find the occupations with maximum and minimum number of new loan applications

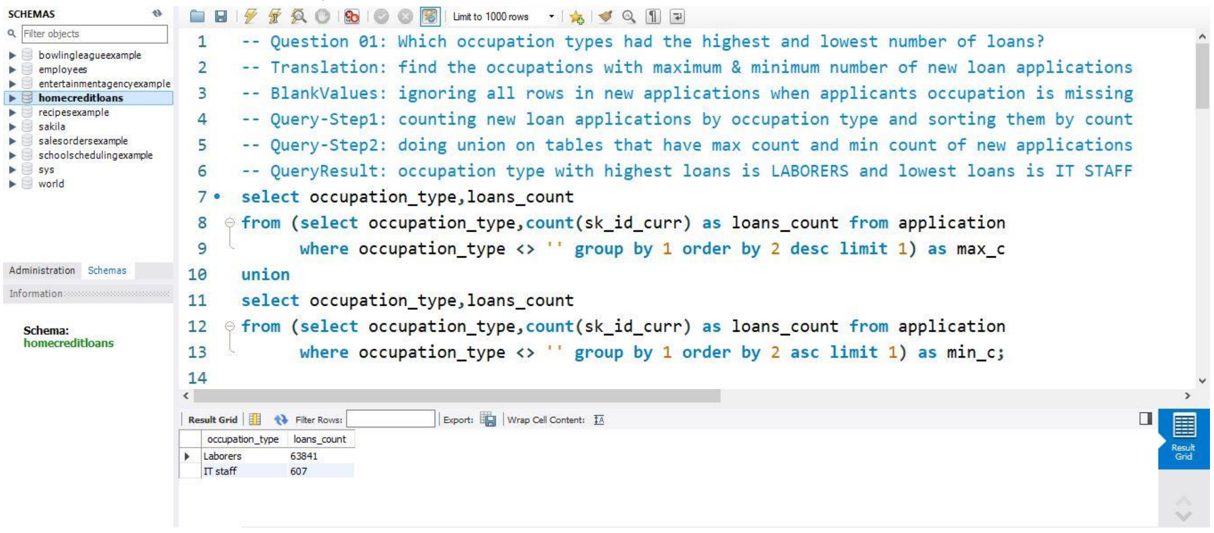
Notes/Comments About SQL Query and Results (Include # of Rows in Result)

Blank Values: ignoring all rows in new applications when an applicant's occupation is missing
Query Step 1: counting new loan applications by occupation type and sorting them by count
Query Step 2: doing union on tables that have max count and min count of new applications
Query Result: occupation type with highest loans is LABORERS and lowest loans is IT STAFF



occupation_type	loans_count
Laborers	63841
IT staff	607

Screen Shot of SQL Query and Results



```sql
-- Question 01: Which occupation types had the highest and lowest number of loans?
-- Translation: find the occupations with maximum & minimum number of new loan applications
-- BlankValues: ignoring all rows in new applications when applicants occupation is missing
-- Query-Step1: counting new loan applications by occupation type and sorting them by count
-- Query-Step2: doing union on tables that have max count and min count of new applications
-- QueryResult: occupation type with highest loans is LABORERS and lowest loans is IT STAFF
select occupation_type,loans_count
from (select occupation_type,count(sk_id_curr) as loans_count from application
      where occupation_type <> '' group by 1 order by 2 desc limit 1) as max_c
union
select occupation_type,loans_count
from (select occupation_type,count(sk_id_curr) as loans_count from application
      where occupation_type <> '' group by 1 order by 2 asc limit 1) as min_c;
```

occupation_type	loans_count
Laborers	63841
IT staff	607

Query 2

Question

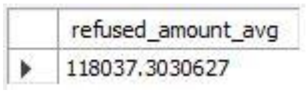
What was the average loan amount denied for Consumer Loans?

Translation

Find the average of amounts applied as consumer loan and got refused

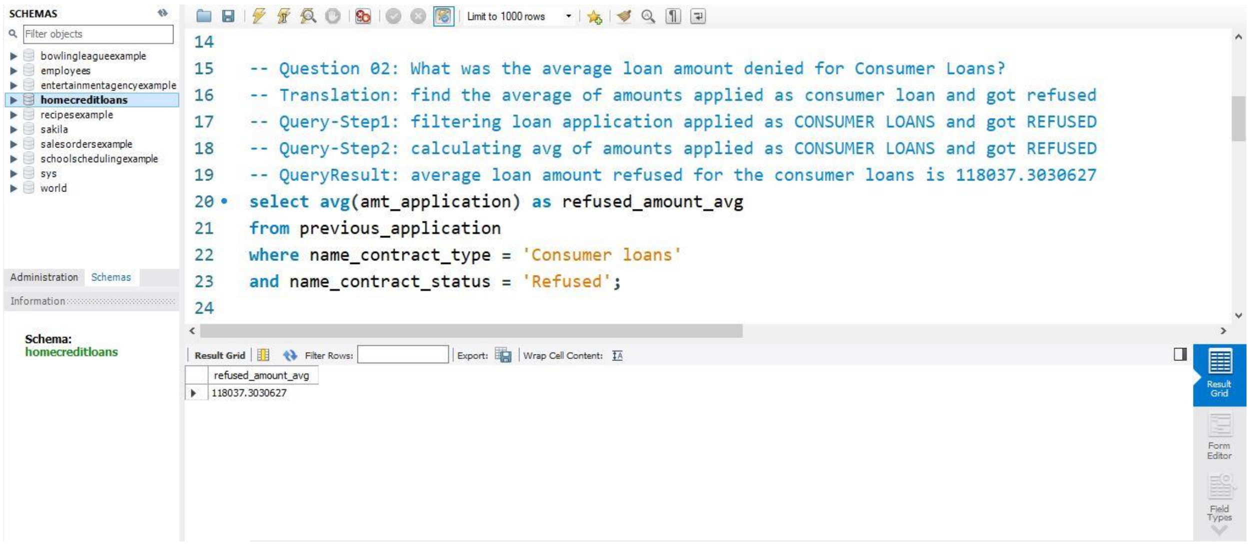
Notes/Comments About SQL Query and Results (Include # of Rows in Result)

Query Step 1: filtering loan applications applied as CONSUMER LOANS and were REFUSED
Query Step 2: calculating avg of amounts applied as CONSUMER LOANS and got REFUSED
Query Result: average loan amount for all the refused consumer loans is 118037.3030627



	refused_amount_avg
▶	118037.3030627

Screen Shot of SQL Query and Results



```
-- Question 02: What was the average loan amount denied for Consumer Loans?
-- Translation: find the average of amounts applied as consumer loan and got refused
-- Query-Step1: filtering loan application applied as CONSUMER LOANS and got REFUSED
-- Query-Step2: calculating avg of amounts applied as CONSUMER LOANS and got REFUSED
-- QueryResult: average loan amount refused for the consumer loans is 118037.3030627
select avg(amt_application) as refused_amount_avg
from previous_application
where name_contract_type = 'Consumer loans'
and name_contract_status = 'Refused';
```

	refused_amount_avg
▶	118037.3030627

Query 3

Question

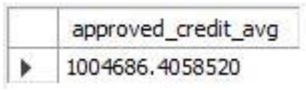
What was the average Cash loan amount approved for Married applicants with higher education?

Translation

Find the average total credit approved as cash loan for married applicants with higher education

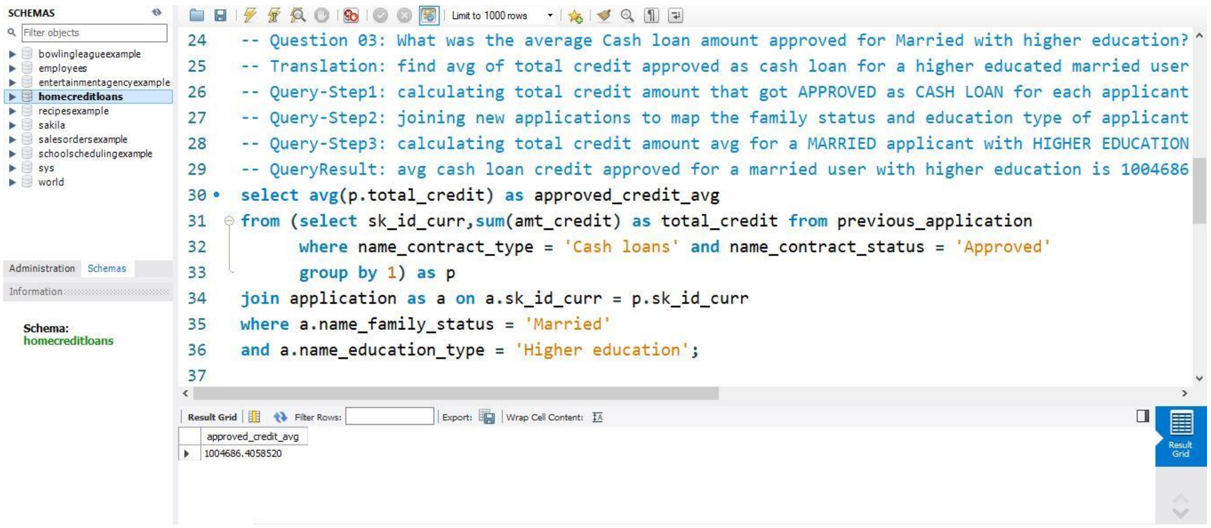
Notes/Comments About SQL Query and Results (Include # of Rows in Result)

Query Step 1: calculating total credit amount that got APPROVED as CASH LOAN for each applicant
Query Step 2: joining new applications to identify the applicant's family status and education type
Query Step 3: calculating total credit amount avg for MARRIED applicant with HIGHER EDUCATION
Query Result: average cash loan credit approved for married one with higher education is 1004686



	approved_credit_avg
▶	1004686.4058520

Screen Shot of SQL Query and Results



```
-- Question 03: What was the average Cash loan amount approved for Married with higher education?
-- Translation: find avg of total credit approved as cash loan for a higher educated married user
-- Query-Step1: calculating total credit amount that got APPROVED as CASH LOAN for each applicant
-- Query-Step2: joining new applications to map the family status and education type of applicant
-- Query-Step3: calculating total credit amount avg for a MARRIED applicant with HIGHER EDUCATION
-- QueryResult: avg cash loan credit approved for a married user with higher education is 1004686
select avg(p.total_credit) as approved_credit_avg
from (select sk_id_curr,sum(amt_credit) as total_credit from previous_application
      where name_contract_type = 'Cash loans' and name_contract_status = 'Approved'
      group by 1) as p
join application as a on a.sk_id_curr = p.sk_id_curr
where a.name_family_status = 'Married'
and a.name_education_type = 'Higher education';
```

	approved_credit_avg
▶	1004686.4058520

Query 5

Question

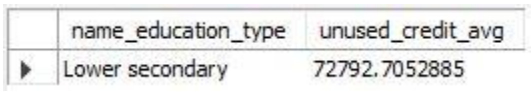
Applicants of which education type had the lowest average unused credit?

Translation

Find the education type with lowest average of total amount credits with unused offers

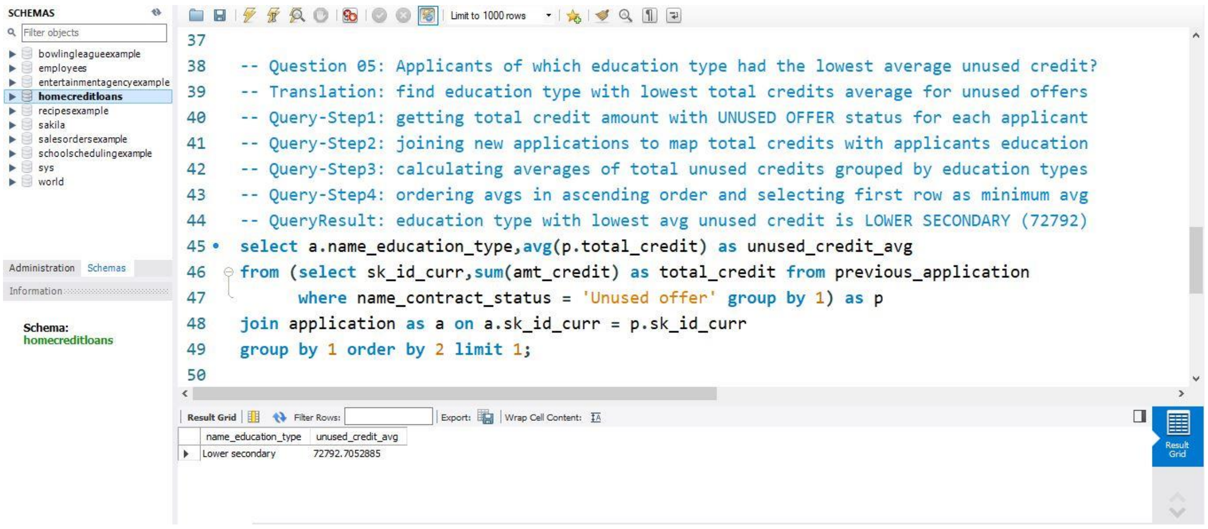
Notes/Comments About SQL Query and Results (Include # of Rows in Result)

Query Step 1: getting total credit amount of each applicant with UNUSED OFFER status
Query Step 2: joining new applications to map total credits with education of applicant
Query Step 3: calculating averages of total unused credits grouped by education types
Query Step 4: ordering averages in ascending order and selecting first row as minimum
Query Result: education type with lowest average unused credit is LOWER SECONDARY



	name_education_type	unused_credit_avg
▶	Lower secondary	72792.7052885

Screen Shot of SQL Query and Results



```
-- Question 05: Applicants of which education type had the lowest average unused credit?
-- Translation: find education type with lowest total credits average for unused offers
-- Query-Step1: getting total credit amount with UNUSED OFFER status for each applicant
-- Query-Step2: joining new applications to map total credits with applicants education
-- Query-Step3: calculating averages of total unused credits grouped by education types
-- Query-Step4: ordering avgs in ascending order and selecting first row as minimum avg
-- QueryResult: education type with lowest avg unused credit is LOWER SECONDARY (72792)
select a.name_education_type,avg(p.total_credit) as unused_credit_avg
from (select sk_id_curr,sum(amt_credit) as total_credit from previous_application
     where name_contract_status = 'Unused offer' group by 1) as p
join application as a on a.sk_id_curr = p.sk_id_curr
group by 1 order by 2 limit 1;
```

	name_education_type	unused_credit_avg
▶	Lower secondary	72792.7052885

Query 6

Question

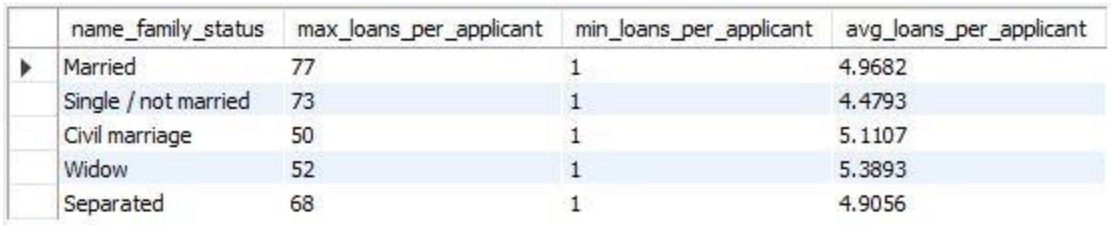
What was the max, min and avg number of loan applications made by an applicant in each family status?

Translation

Find the maximum, minimum and average of old applications counts for each applicant by family status

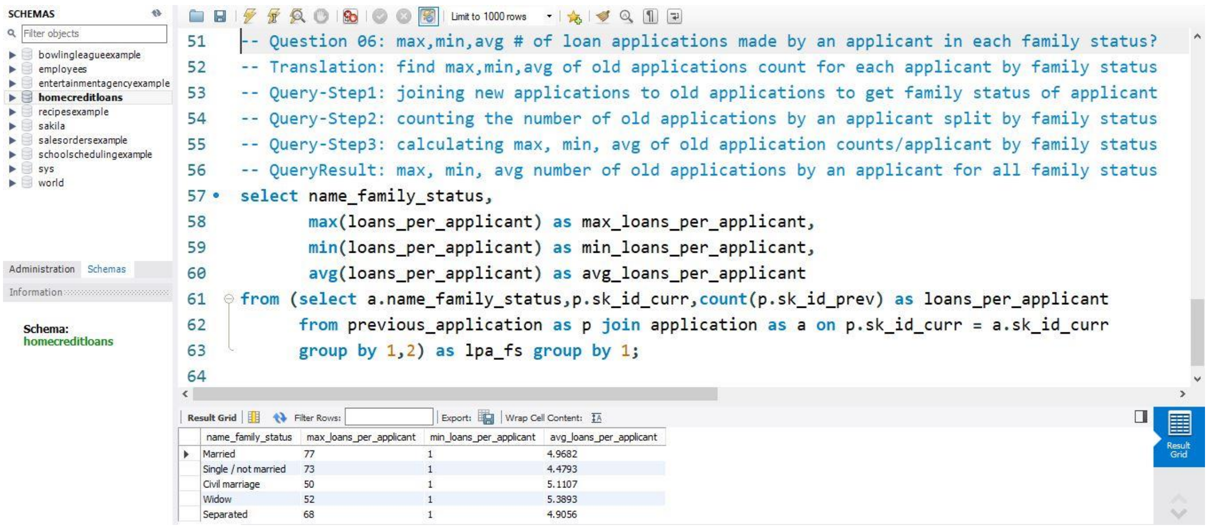
Notes/Comments About SQL Query and Results (Include # of Rows in Result)

Query Step 1: joining new applications to old applications to get family status of applicants
Query Step 2: counting the number of old applications by an applicant split by family status
Query Step 3: calculating max, min, avg of old application counts/applicant by family status
Query Result: max, min, avg number of old applications by an applicant for all family status



name_family_status	max_loans_per_applicant	min_loans_per_applicant	avg_loans_per_applicant
Married	77	1	4.9682
Single / not married	73	1	4.4793
Civil marriage	50	1	5.1107
Widow	52	1	5.3893
Separated	68	1	4.9056

Screen Shot of SQL Query and Results



```
-- Question 06: max,min,avg # of loan applications made by an applicant in each family status?
-- Translation: find max,min,avg of old applications count for each applicant by family status
-- Query-Step1: joining new applications to old applications to get family status of applicant
-- Query-Step2: counting the number of old applications by an applicant split by family status
-- Query-Step3: calculating max, min, avg of old application counts/applicant by family status
-- QueryResult: max, min, avg number of old applications by an applicant for all family status
select name_family_status,
       max(loans_per_applicant) as max_loans_per_applicant,
       min(loans_per_applicant) as min_loans_per_applicant,
       avg(loans_per_applicant) as avg_loans_per_applicant
from (select a.name_family_status,p.sk_id_curr,count(p.sk_id_prev) as loans_per_applicant
      from previous_application as p join application as a on p.sk_id_curr = a.sk_id_curr
      group by 1,2) as lpa_fs group by 1;
```

name_family_status	max_loans_per_applicant	min_loans_per_applicant	avg_loans_per_applicant
Married	77	1	4.9682
Single / not married	73	1	4.4793
Civil marriage	50	1	5.1107
Widow	52	1	5.3893
Separated	68	1	4.9056

Data Review for MongoDB

Assumptions/Notes About Collections, Attributes and Relationships between Collections

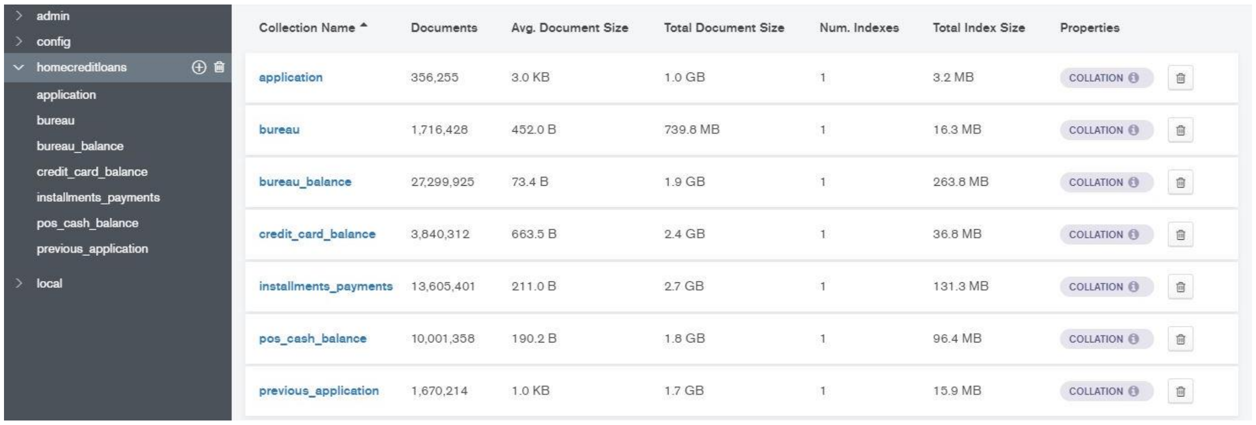
- application {train|test}.csv
    - This is the main table, broken into two files for Train (with TARGET) and Test (without TARGET).
    - Static data for all applications. One row represents one loan in our data sample.
    - Every loan has its own row and is identified by the feature SK_ID_CURR. The training application data comes with the TARGET indicating 0: the loan was repaid or 1: the loan was not repaid.
- bureau.csv
    - All client's previous credits provided by other financial institutions that were reported to Credit Bureau (for clients who have a loan in our sample).
    - For every loan in our sample, there are as many rows as number of credits the client had in Credit Bureau before the application date.
    - Each previous credit has its own row in bureau, but one loan in the application data can have multiple previous credits.
- bureau_balance.csv
    - Monthly balances of previous credits in Credit Bureau.
    - This table has one row for each month of history of every previous credit reported to Credit Bureau - i.e. the table has (number of loans in sample * number of relative previous credits * number of months where we have some history observable for the previous credits) rows.
    - Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.
- POS_CASH_balance.csv
    - Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.
    - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample - i.e. the table has (number of loans in sample * number of relative previous credits * number of months in which we have some history observable for the previous credits) rows.
    - Each row is one month of a previous point of sale or cash loan, and a single previous loan can have many rows.
- credit_card_balance.csv
    - Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.
    - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample - i.e. the table has (number of loans in sample * number of relative previous credit cards * number of months where we have some history observable for the previous credit card) rows.
    - Each row is one month of a credit card balance, and a single credit card can have many rows.
- previous_application.csv
    - All previous applications for Home Credit loans of clients who have loans in our sample.
    - There is one row for each previous application related to loans in our data sample.
    - Each current loan in the application data can have multiple previous loans. Each previous application has one row and is identified by the feature SK_ID_PREV.
- installments_payments.csv
    - Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.
    - There is a) one row for every payment that was made plus b) one row each for missed payment.
    - One row is equivalent to one payment of one installment OR one installment corresponding to one payment of one previous Home Credits credit related to loans in our sample.

Physical Mongo Database

Assumptions/Notes About Data Set

Merged application test & application train datasets with the target column as blank for test dataset
Many columns in almost all the tables have missing values and are uploaded into database as blanks

Screen shot of Physical Database objects (Database, Collections and Attributes)



Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
application	356,255	3.0 KB	1.0 GB	1	3.2 MB	COLLATION
bureau	1,716,428	452.0 B	739.8 MB	1	16.3 MB	COLLATION
bureau_balance	27,299,925	73.4 B	1.9 GB	1	263.8 MB	COLLATION
credit_card_balance	3,840,312	663.5 B	2.4 GB	1	36.8 MB	COLLATION
installments_payments	13,605,401	211.0 B	2.7 GB	1	131.3 MB	COLLATION
pos_cash_balance	10,001,358	190.2 B	1.8 GB	1	96.4 MB	COLLATION
previous_application	1,670,214	1.0 KB	1.7 GB	1	15.9 MB	COLLATION

Data in the Database

Collection Name	Relationships with Other Collections (if any)	# of Rows in Table
application	indexed on sk_id_curr	356,255
previous_application	indexed on sk_id_prev sk_id_curr references sk_id_curr from application	1,670,214
bureau	indexed on sk_id_bureau sk_id_curr references sk_id_curr from application	1,716,428
bureau_balance	indexed on sk_id_bureau, months_balance sk_id_bureau references sk_id_bureau from bureau	27,299,925
pos_cash_balance	indexed on sk_id_prev, months_balance sk_id_curr references sk_id_curr from application sk_id_prev references sk_id_prev from previous_application	10,001,358
credit_card_balance	indexed on sk_id_prev, months_balance sk_id_curr references sk_id_curr from application sk_id_prev references sk_id_prev from previous_application	3,840,312
installments_payments	indexed on sk_id_payment sk_id_curr references sk_id_curr from application sk_id_prev references sk_id_prev from previous_application	13,605,401

MongoDB Queries/Code

Query 1

Question

Which occupation types had the highest and lowest number of loans?

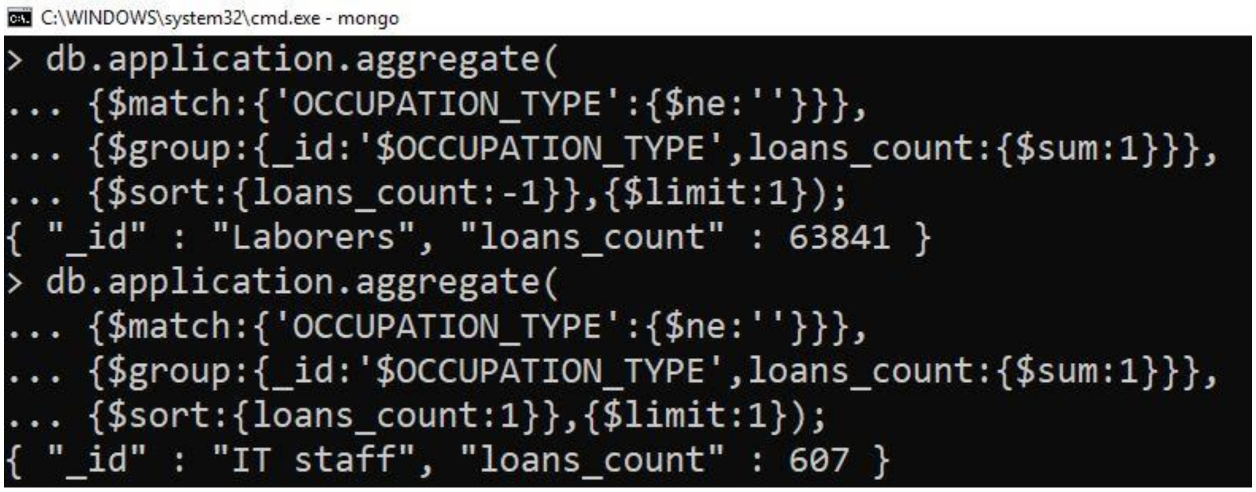
Translation

Find the occupations with maximum and minimum number of new loan applications

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Blank Values: ignoring all rows in new applications when an applicant's occupation is missing
Query Step 1: counting loan applications by occupation type and sorting in descending order
Query Step 2: limiting result to top row that has occupation with maximum number of loans
Query Step 3: counting loan applications by occupation type and sorting in ascending order
Query Step 4: limiting result to top row that has occupation with minimum number of loans
Query Result: occupation type with highest loans is LABORERS and lowest loans is IT STAFF

Screen Shot of MongoDB Query/Code and Results



C:\WINDOWS\system32\cmd.exe - mongo

```
> db.application.aggregate(
... {$match:{'OCCUPATION_TYPE':{$ne:''}}},
... {$group:{_id:'$OCCUPATION_TYPE',loans_count:{$sum:1}}},
... {$sort:{loans_count:-1}},{$limit:1});
{ "_id" : "Laborers", "loans_count" : 63841 }
> db.application.aggregate(
... {$match:{'OCCUPATION_TYPE':{$ne:''}}},
... {$group:{_id:'$OCCUPATION_TYPE',loans_count:{$sum:1}}},
... {$sort:{loans_count:1}},{$limit:1});
{ "_id" : "IT staff", "loans_count" : 607 }
```

Query 2

Question

What was the average loan amount denied for Consumer Loans?

Translation

Find the average of amounts applied as consumer loan and got refused

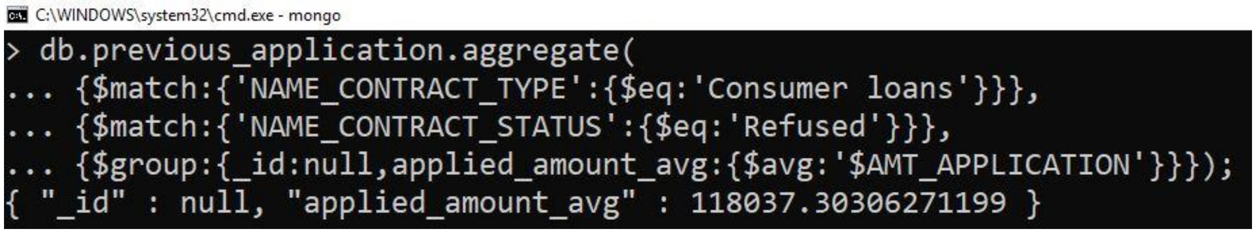
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Query Step 1: filtering loan applications applied as CONSUMER LOANS and were REFUSED

Query Step 2: calculating avg of amounts applied as CONSUMER LOANS and got REFUSED

Query Result: average loan amount for all the refused consumer loans is 118037.3030627

Screen Shot of MongoDB Query/Code and Results



```
> db.previous_application.aggregate(
... {$match:{'NAME_CONTRACT_TYPE':{$eq:'Consumer loans'}}},
... {$match:{'NAME_CONTRACT_STATUS':{$eq:'Refused'}}},
... {$group:{_id:null,applied_amount_avg:{$avg:'$AMT_APPLICATION'}}});
{ "_id" : null, "applied_amount_avg" : 118037.30306271199 }
```

Query 3

Question

What was the average Cash loan amount approved for Married applicants with higher education?

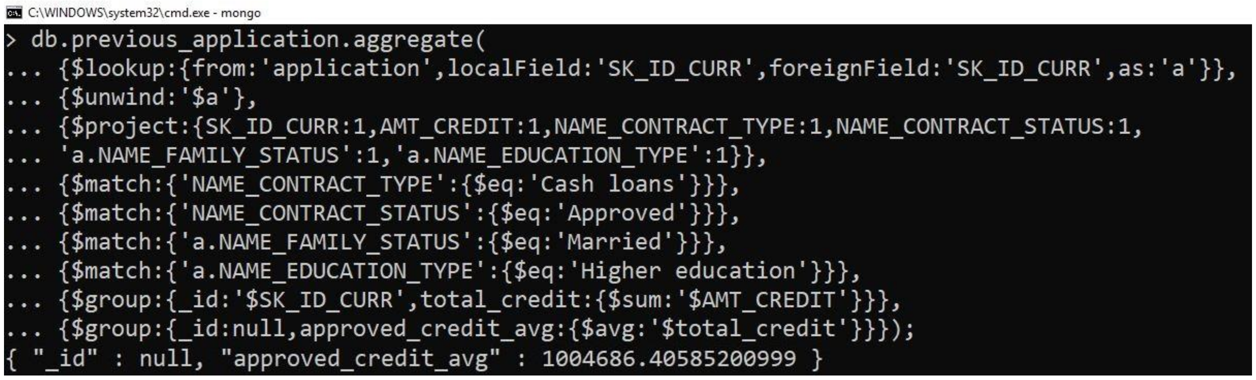
Translation

Find the average total credit approved as cash loan for married applicants with higher education

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Query Step 1: joining new applications to identify the applicant's family status and education type
Query Step 2: filtering all the old loan applications applied as CASH LOANS which were APPROVED
Query Step 3: filtering all loan application applied by MARRIED applicant with HIGHER EDUCATION
Query Step 4: calculating the total credit amount applied by an applicant with the above attributes
Query Step 5: calculating avg credit APPROVED as CASH LOANS for MARRIED, HIGHER EDUCATION
Query Result: average cash loan credit approved for married one with higher education is 1004686

Screen Shot of MongoDB Query/Code and Results



C:\WINDOWS\system32\cmd.exe - mongo

```
> db.previous_application.aggregate(
... {$lookup:{from:'application',localField:'SK_ID_CURR',foreignField:'SK_ID_CURR',as:'a'}},
... {$unwind:'$a'},
... {$project:{SK_ID_CURR:1,AMT_CREDIT:1,NAME_CONTRACT_TYPE:1,NAME_CONTRACT_STATUS:1,
... 'a.NAME_FAMILY_STATUS':1,'a.NAME_EDUCATION_TYPE':1}},
... {$match:{'NAME_CONTRACT_TYPE':{$eq:'Cash loans'}}},
... {$match:{'NAME_CONTRACT_STATUS':{$eq:'Approved'}}},
... {$match:{'a.NAME_FAMILY_STATUS':{$eq:'Married'}}},
... {$match:{'a.NAME_EDUCATION_TYPE':{$eq:'Higher education'}}},
... {$group:{_id:'$SK_ID_CURR',total_credit:{$sum:'$AMT_CREDIT'}}},
... {$group:{_id:null,approved_credit_avg:{$avg:'$total_credit'}}});
{ "_id" : null, "approved_credit_avg" : 1004686.40585200999 }
```

Query 5

Question

Applicants of which education type had the lowest average unused credit?

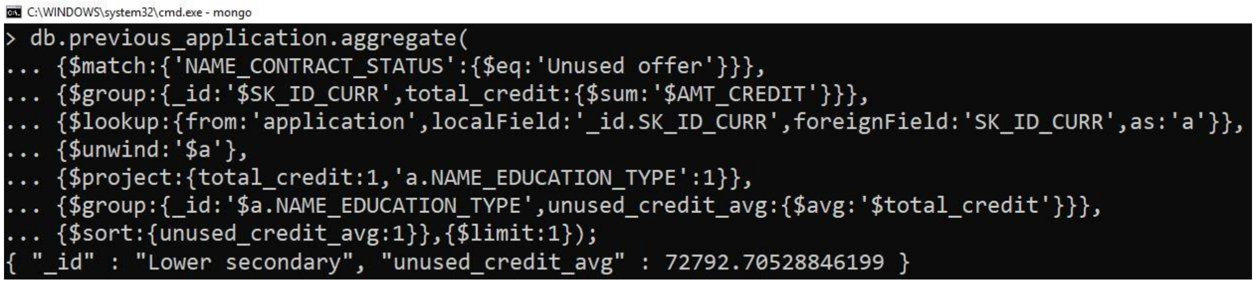
Translation

Find the education type with lowest average of total amount credits with unused offers

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Query Step 1: filtering the old loan applications with contract status as UNUSED OFFER
Query Step 2: getting total credit amount of each applicant with UNUSED OFFER status
Query Step 3: joining new applications to identify the education types of the applicant
Query Step 4: calculating averages of total unused credits grouped by education types
Query Step 5: ordering average in ascending order and selecting the top row as lowest
Query Result: education type with lowest average unused credit is LOWER SECONDARY

Screen Shot of MongoDB Query/Code and Results



```
> db.previous_application.aggregate(
... {$match:{'NAME_CONTRACT_STATUS':{$eq:'Unused offer'}}},
... {$group:{_id:'$SK_ID_CURR',total_credit:{$sum:'$AMT_CREDIT'}}},
... {$lookup:{from:'application',localField:'_id.SK_ID_CURR',foreignField:'SK_ID_CURR',as:'a'}},
... {$unwind:'$a'},
... {$project:{total_credit:1,'a.NAME_EDUCATION_TYPE':1}},
... {$group:{_id:'$a.NAME_EDUCATION_TYPE',unused_credit_avg:{$avg:'$total_credit'}}},
... {$sort:{unused_credit_avg:1}},{$limit:1});
{ "_id" : "Lower secondary", "unused_credit_avg" : 72792.70528846199 }
```

Query 6

Question

What was the max, min and avg number of loan applications made by an applicant in each family status?

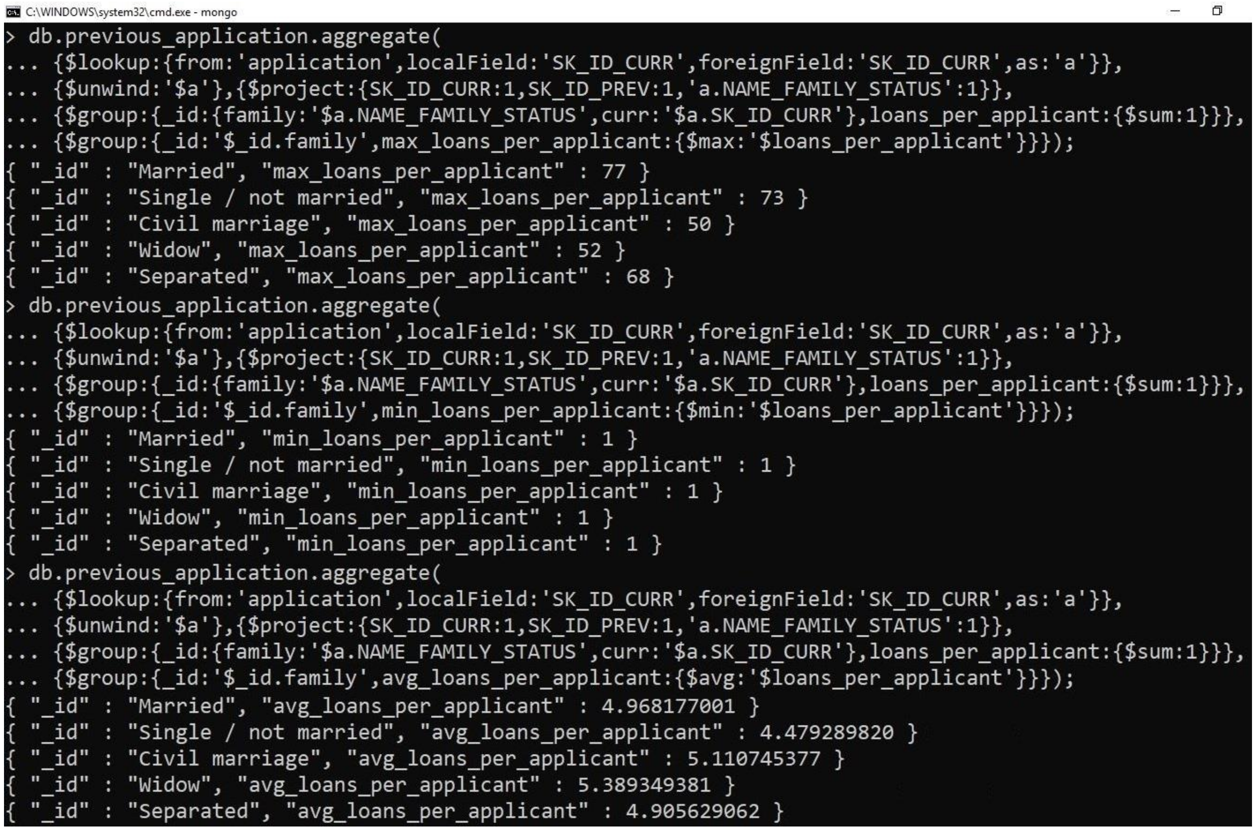
Translation

Find the maximum, minimum and average of old applications counts for each applicant by family status

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Query Step 1: joining new applications to old applications to get family status of applicants
Query Step 2: counting the number of old applications by an applicant split by family status
Query Step 3: calculating max, min, avg of old application counts/applicant by family status
Query Result: max, min, avg number of old applications by an applicant for all family status

Screen Shot of MongoDB Query/Code and Results

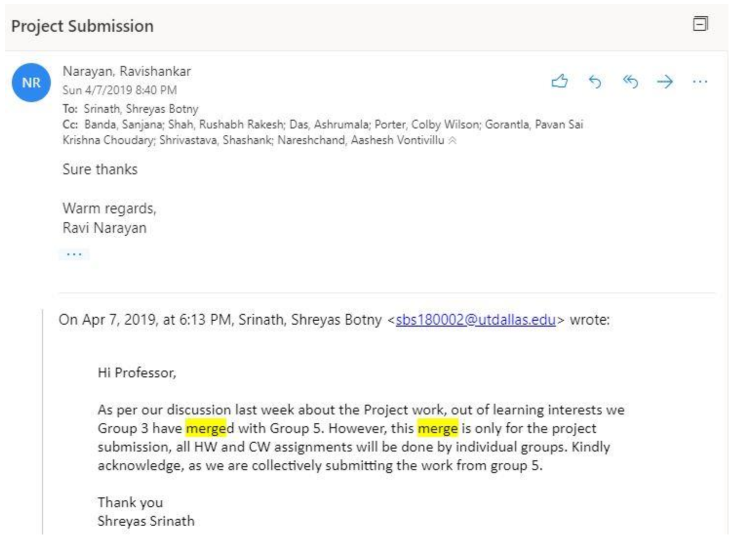


```
C:\WINDOWS\system32\cmd.exe - mongo
> db.previous_application.aggregate(
... {$lookup:{from:'application',localField:'SK_ID_CURR',foreignField:'SK_ID_CURR',as:'a'}},
... {$unwind:'$a'},{$project:{SK_ID_CURR:1,SK_ID_PREV:1,'a.NAME_FAMILY_STATUS':1}},
... {$group:{_id:{family:'$a.NAME_FAMILY_STATUS',curr:'$a.SK_ID_CURR'},loans_per_applicant:{$sum:1}}},
... {$group:{_id:'$_id.family',max_loans_per_applicant:{$max:'$loans_per_applicant'}}});
{ "_id" : "Married", "max_loans_per_applicant" : 77 }
{ "_id" : "Single / not married", "max_loans_per_applicant" : 73 }
{ "_id" : "Civil marriage", "max_loans_per_applicant" : 50 }
{ "_id" : "Widow", "max_loans_per_applicant" : 52 }
{ "_id" : "Separated", "max_loans_per_applicant" : 68 }
> db.previous_application.aggregate(
... {$lookup:{from:'application',localField:'SK_ID_CURR',foreignField:'SK_ID_CURR',as:'a'}},
... {$unwind:'$a'},{$project:{SK_ID_CURR:1,SK_ID_PREV:1,'a.NAME_FAMILY_STATUS':1}},
... {$group:{_id:{family:'$a.NAME_FAMILY_STATUS',curr:'$a.SK_ID_CURR'},loans_per_applicant:{$sum:1}}},
... {$group:{_id:'$_id.family',min_loans_per_applicant:{$min:'$loans_per_applicant'}}});
{ "_id" : "Married", "min_loans_per_applicant" : 1 }
{ "_id" : "Single / not married", "min_loans_per_applicant" : 1 }
{ "_id" : "Civil marriage", "min_loans_per_applicant" : 1 }
{ "_id" : "Widow", "min_loans_per_applicant" : 1 }
{ "_id" : "Separated", "min_loans_per_applicant" : 1 }
> db.previous_application.aggregate(
... {$lookup:{from:'application',localField:'SK_ID_CURR',foreignField:'SK_ID_CURR',as:'a'}},
... {$unwind:'$a'},{$project:{SK_ID_CURR:1,SK_ID_PREV:1,'a.NAME_FAMILY_STATUS':1}},
... {$group:{_id:{family:'$a.NAME_FAMILY_STATUS',curr:'$a.SK_ID_CURR'},loans_per_applicant:{$sum:1}}},
... {$group:{_id:'$_id.family',avg_loans_per_applicant:{$avg:'$loans_per_applicant'}}});
{ "_id" : "Married", "avg_loans_per_applicant" : 4.968177001 }
{ "_id" : "Single / not married", "avg_loans_per_applicant" : 4.479289820 }
{ "_id" : "Civil marriage", "avg_loans_per_applicant" : 5.110745377 }
{ "_id" : "Widow", "avg_loans_per_applicant" : 5.389349381 }
{ "_id" : "Separated", "avg_loans_per_applicant" : 4.905629062 }
```

Project Contributors

Activity	Pavan Sai Krishna Gorantla	Aashesh Nareshchand	Shashank Srivastava	Colby Porter
Prepared Data Model			X	X
Created Physical Database			X	X
Loaded Data into Database	X	X		
Wrote SQL Queries	X	X		
Prepared Mongo Database			X	X
Loaded Data into Mongo DB	X	X		
Wrote Mongo Queries	X	X		
Prepared Report	X	X		
Reviewed Report			X	X

With professor's permission, Group #3 members requested us to be included into our Group #5



Project Submission

Narayan, Ravishankar
Sun 4/7/2019 8:40 PM
To: Srinath, Shreyas Botny
Cc: Banda, Sanjana; Shah, Rushabh Rakesh; Das, Ashrumala; Porter, Colby Wilson; Gorantla, Pavan Sai Krishna Choudary; Shrivastava, Shashank; Nareshchand, Aashesh Vontivillu

Sure thanks

Warm regards,
Ravi Narayan

...

On Apr 7, 2019, at 6:13 PM, Srinath, Shreyas Botny <sbs180002@utdallas.edu> wrote:

Hi Professor,

As per our discussion last week about the Project work, out of learning interests we Group 3 have merged with Group 5. However, this merge is only for the project submission, all HW and CW assignments will be done by individual groups. Kindly acknowledge, as we are collectively submitting the work from group 5.

Thank you
Shreyas Srinath

Activity	Shreyas Botny Srinath	Rushabh Rakesh Shah	Ashrumala Das
Prepared Data Model	X	X	X
Created Physical Database	X	X	X
Prepared Mongo Database	X	X	X