جامعة جدة
University of Jeddah

CCCS-215 Introduction to Database 14172

Bank management system

By :
Ohood Tariq Alsheikh

&

Nawal Awad Algrni

&

Raghad Fawzi Alsyed

&

Joud Tarek Bawajeeh

Supervised by:

Dr. Ahlem Ben Hassine

Date: 5 November 2020

Team members :

Name of members	ID of members	Email of email
Ohood Tariq Alsheikh (The leader)	1910867	1910867@uj.edu.sa
Nawal Awad Algrni	1913763	Nalqarni0036.stu@uj.edu.sa
Raghad Fawzi Alsyed	1911352	1911352@uj.edu.sa
Joud Tarek	1911327	1911327@uj.edu.sa

Project title : Bank management system

The bank offers many services to its clients . Provides a service of opening a checking account or providing deposit services, ATM cards, loans and other services. The bank suffers from difficulty in managing the loan service, as the loan service is still provided manually. When the client submits a request for a loan, the employee manually checks whether this client has the right to obtain a loan or not, which leads to the customer visiting the bank several times and increasing the pressure On the employe e. The solution is to create a system that automatically determines whether or not a customer can obtain the loan without having to visit the bank .

Project schedule:

tasks	due date	responsible member
Phase 1: DB Analysis	4 October 2020	Ohood Tariq Alsheikh
Phase 2: DB Design : Part A (ER Model)	18 October 2020	Nawal Awad Algrni
Phase 2: DB Design : Part B (Normalization and Mapping)	1 November 2020	Joud Tarek
Phase 3 : DB Implementation & testing	29 November 2020	Raghad Fawzi Alsyed

Chapter 1 : DB Analysis

1.1 Organization background

The Bank serves a variety of clients and provides suitable services for each category. Banking services such as personal banking services, which include opening an account whether it is a current account type or a savings account .It also provides personal loans and financial deposits, as well as international transfer service and credit cards. The bank also offer the opportunity to withdraw and deposit funds in a number of ways, such as going to the bank's headquarters, using an ATM or accessing the bank's website.

The bank divides its clients in terms of account balance into three categories: low balance category ($0-1000), median balance category ($10,001-$25,000) and high balance category ($25,001- and above). Each category has its own services. Like the atm card service, the low-balance category gets a copper card , the middle balance category gets a silver card and the high balance category gets a gold card.

The Bank focuses on the loan service. There are six types of loans the bank offers to its clients : personal loans, mortgages , credit card loans, study loans, investment loans, travel and tourism loans. Personal loans are a means of guarantee for individuals that are used in personal matters and are for example for purchase and payment purposes. A mortgage is a loan that enables the client to borrow money to buy a house or any other property, and his ownership of this property is a guarantee of the loan. Credit card loans provide the owner with access to money, often used in purchases, every time a purchase is made through credit cards, the customer adding a sum of debt on his account with the bank that issued the card. study loans are loans for individuals wishing to complete their university studies. Investment loans are loans that are repaid over a period of 30 years, perhaps more. This type of loan is called long-term loans and is used when the client wishes to create a large business. Travel and tourism loans come in order to meet the large expenses required by travel ranging from airline tickets, hotel reservations, transportation costs, etc.

1.2 Current situation/ problems

The bank has difficulty managing the loan service. where the loan service still manually offered . When the customer comes to apply for a loan, the employee manually checks whether the customer has the right to receive the loan or not. This leads to increased pressure on employees, loss of time and effort, and increased number of customer visits to the bank, so the process takes a lot of time. This, of course, may cause customer dissatisfaction, which makes him consider going to another bank. The solution is to create a management system for the loan service. Where the client and he is in his place , opens the application of the bank and goes to the section of loans and choose the type of loan that he wants and submit on it , on the other hand the system accepts the client's request and automatically checks that the client has the validity to get the loan or not and shows the result to the employee. The employee in turn communicates with the client to inform him about the result.

1.3 The business rules of the system

1- Every client in the bank must have a running account .
2- Client can’t have another loan unless she/he paid what they owe .
3- Clients must provide a letter from the employer that includes the job type, date of employment.
4- The minimum salary is 3000 S.R to extract a loan .
5- The maximum age is 50 years or at least 18 years old .
6- In Mortgage loans if the client doesn’t make a monthly payment the bank can sell the home and recoup its money .

1.4 DB application/data requirements

The system to be developed is one that aims to reduce pressure on employees and achieve social distancing according to the current conditions. The system works on serving the customer applying for a loan, whereby the customer is required to provide all the data required to apply for a loan, such as his personal data, the amount submitted for his request and the amount of his monthly income that will be stored in a database and determine the type of loan, after that the system will submit the request to the bank’s employees , Who in turn send approval or rejection .

1.5 The outputs of the system

a calculator to calculate the loan clients can get based on their salary. Statistic show how many clients are asking for a loan per month and what the average age of those clients and the sector they’re working in .To help the system find the most category of clients who asked for a loan to target them with commercial advertisements and attract new clients . The system should help to match the clients with the rules automatically .

1.6 Collect the information

We adopted several things that helped us collect a lot of information, but before we started collecting information, we asked several questions, including: Does the bank need a database? What are the problems facing the bank? What problems do employees face within the bank? What do customers and employees need? What are the things that the bank needs to help reduce problems and risks?

We adopted some of these questions to help us in the process of searching and exploring information.

These questions helped us to find serious ideas and valuable information, and we also adopted in the process of searching for information the brainstorming strategy, so through the process of brainstorming we were able to start storing ideas, arranging them, and classifying problems and solutions.

And we were also able to find some useful information from some internet sites that may be useful for the process of collecting the information needed by the team and the project. We found some information that is useful in creating a protected database and classifying categories and how to solve problems such as: problems of bank loans of all kinds and method Solving these problems, defining types of bank loans, defining types of investments, and other matters . It is these sites such as:

1- https://mawdoo3.com/
2- https://www.masralyoum.net/
3- https://www.alahli.com/ar-sa/personal-banking/islamicFinance/Pages/Overview_.aspx
4- https://ar.m.wikipedia.org/wiki/

1.7 Data Dictionary

Data	Data Description	Data Type
Account Id	The ID for each account in the bank	Number
Account Type	The Type for each account in the bank	String
IBAN number	The IBAN number for each account in the bank	Number
Password	The Password for account in the bank	String
Opened date	The date the account was opened	Date
Client Id	The ID for each Client in the bank	Number
Client Name	The Name for each Client in the bank	String
Date of Birth	The Date of Birth for each Client in the bank	date
Client age	The age for each Client in the bank	Number
Client nationality	The nationality for each Client in the bank	String
Client address	The address for each Client in the bank	String
Phone Number	The Phone Number for each Client in the bank	Number
Client email	The email for each Client in the bank	String
Income	The amount of Income for each Client in the bank	Number
Employee Id	The ID for each Employee in the bank .	Number
Employee Name	The name for each Employee in the bank .	String
Salary	The amount of Salary for each Employee in the bank	Number
Years of Experience	The Years of Experience for each Employee in the bank .	Number
Employee nationality	The nationality for each Employee in the bank .	String
Date of Birth	The Date of Birth for each Employee in the bank .	Date
Employee age	The age for each Employee in the bank .	Number
Employee gender	The gender for each Employee in the bank .	String
Department	The department in which the employee works	String
Contact number	The Contact number for each Employee in the bank .	Number
Employee email	The email for each Employee in the bank .	String
Employee address	The address for each Employee in the bank .	String
Privilege	Privileges obtained by the employee	String
Loan ID	The ID for each loan that the bank offers it to its Clients	Number
Loan Type	The type for each loan that the bank offers it to its Clients	String
Date of loan	The due date of the loan.	Date
Duration	The period of time the loan is to be paid off	Number
Total Amount	The total loan amount	Number
Remaining Amount	The remaining unpaid amount of the loan	Number

Description	Description of the reason for applying for the loan	String
Status	Current loan status	String
Mortgages	It shows the mortgage and who owns it in the case of a mortgage loan	String
Gold Card	ATM card that is offered to high-income customers	String
Bronze card	ATM card that is offered to low-income customers	String
Silver card	ATM card that is offered to medium-income customers	String
Student Id	In the case of study loans, the customer must provide the student ID	Number
School name	In the case of study loans, the customer must provide the school name	String
Interest	The interest that the bank receives from the customer when opening a savings account	Number
Loan Interest	The interest that the bank receives from the customer when obtaining a loan	Number
Checkbook	The customer receives a check book when opening a current account	String
Balance	The amount of the balance in the account	Number
Branch ID	The ID for each Branch of the bank	Number
Branch name	The name for each Branch of the bank	String
Telephone number	The Telephone number for each Branch of the bank	Number
Location of a branch	The Location for each Branch of the bank	String

Chapter 2 : DB Design

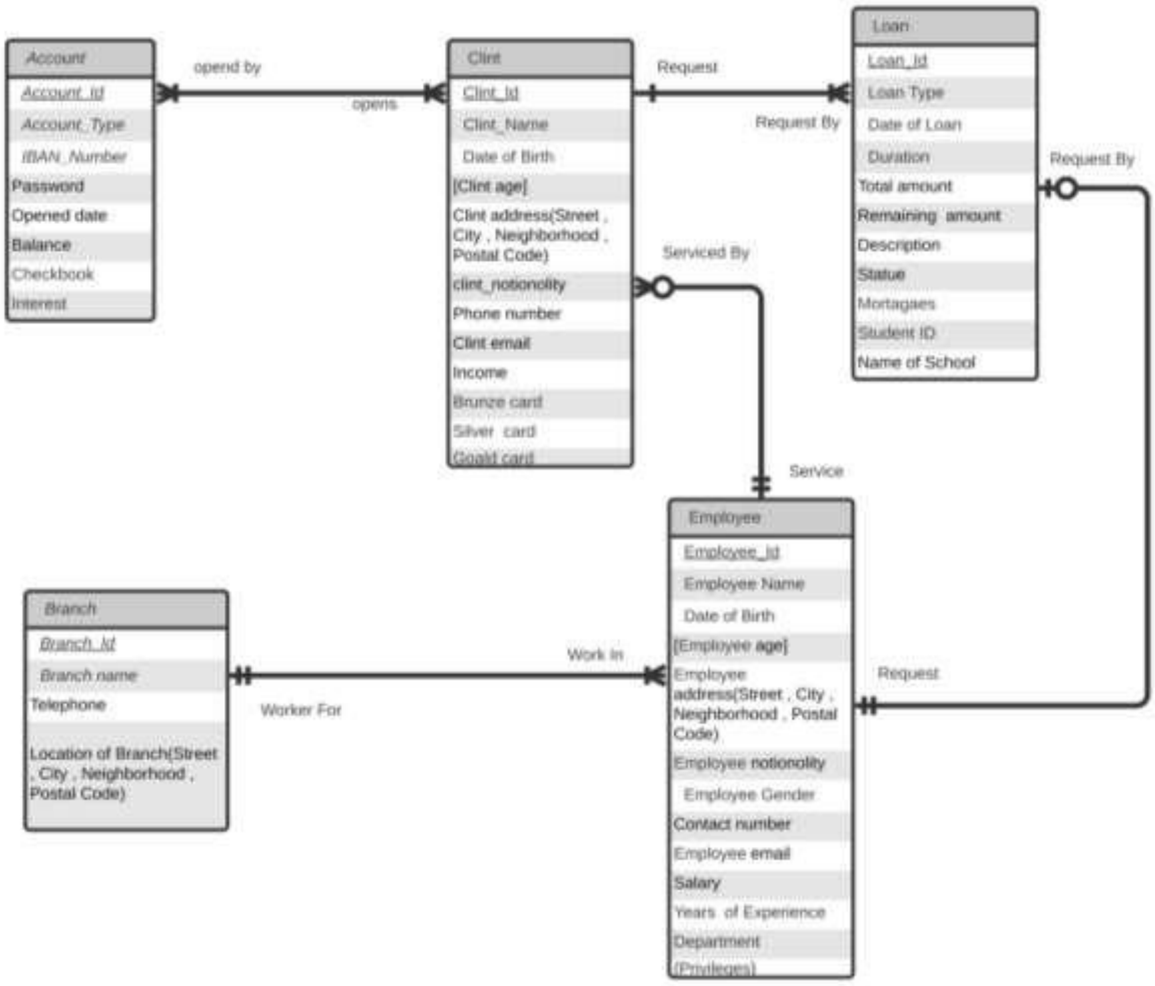
2.1 Part A (ER Model)

First we discussed the bank's data system , then we categorized the data types that were collected and categorized them into entities. Then we found that some Attributes does not apply to all data, so we converted ER to EER To become more clear and understanding .

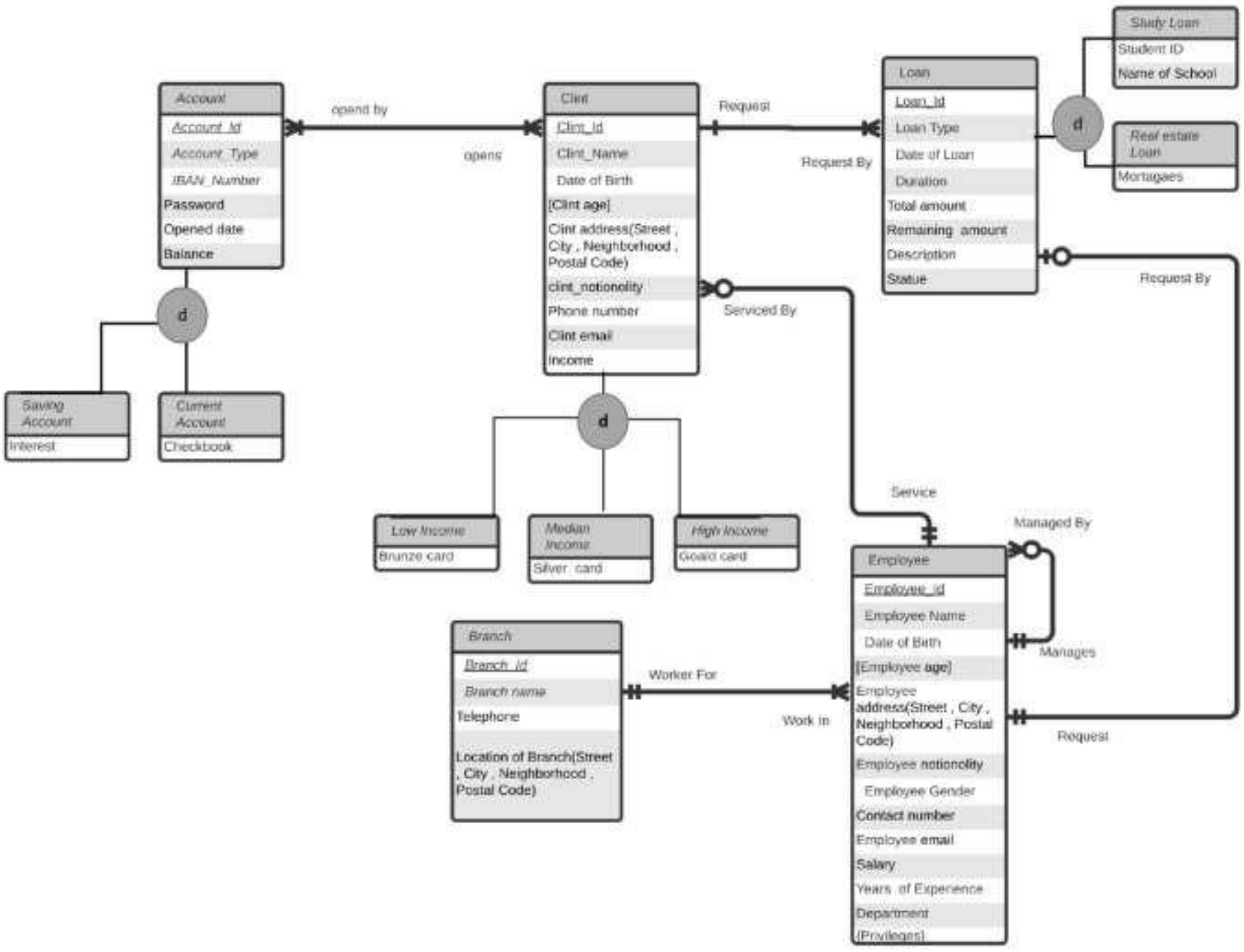
2.1.1 the Brief description about Entities

- Account entity : This entity provides two types of accounts a client can assign . When creating the account it's has : The account's id (primary key) that distinguishes each account from other , password , IBAN number , date of account open and balance.
- Client entity : This entity to obtain client information is the desire to open an account in the bank if it is current or saving Such as: Client ID, Client name , date of birth, Client age, Client nationality , Client address , Phone number , Email and Salary . The primary key will be the customer id .
- Gold card entity : Every high-income Client gets a gold ATM card .
- Silver card entity : Every middle-income Client gets a silver ATM card .
- Bronze card entity : Every low-income Client gets a bronze ATM card .
- Saving account entity : This type of account takes interest for every payment that the client add to account . To enable the customer to collect some amounts within the account and to be opened later by the customer.
- Current account entity : This type of account provides a checkbook for every client .From this account he can perform some operations within the account such as disbursement, deposits, balance disclosure and other operations.
- The loan entity : One of the bank services allow the clients to extract is a loan. Loan Id (pk) , Loan Type , Date of Loan , Duration , Total Amount ,Remaining Amount ,Description , Statue ,Loan interest .
- Real estate loan entity : This loan is ONLY for Saudi clients It will be a mortgage owned by the bank until the client pay the loan .
- Study loan entity : This type ONLY for student wishing to complete their studies or pay their children's education dues.
- Branch entity: It determines which branch the customer and employees are assigned in . With BranchID as a PK , Branch name ,Telephone number ,Location of the Branch (Street , City , neighborhood , Postal Code).
- Employee entity: It determines the supervision between the employee and the client , Employee ID (pk) , Employee Name , Date of birth , Employee age ,Employee nationality ,Employee gender ,Employee address(Street , City , neighborhood , Postal Code) ,Contact number ,Employee email ,Salary ,Years of Experience ,Department ,Privileges

2.1.1 The ER Model



2.1.2 The EER Model



2.2 Part B (Normalization and Mapping) :

2.2.1 : The different used steps to perform the Normalization:

1- In branch relation , it was not at 1NF because it had a composite attribute , which is the location . So we have broken it down into the following attributes: **city, street, neighborhood, postal code**. Then we checked it and did not find partial dependency or transitive dependency . So it is in 3NF .

2- In The employee relation , it has a composite attribute (employee address) and multivalued attribute (privileges) it was not at 1NF . So we have broken down the composite attribute into the following attributes: **city, street, neighborhood, postal code .** and we make a new relation we called it **Employee benefits** and its PK is (Employee ID and privileges). Then we checked it and did not find partial dependency but we find the transitive dependency between The date birth and Age , so we make a new relation To remove of the transitive dependency , we called it **Age of Employee** . The date birth in employee relation is FK from Age of Employee relation . So it become in 3NF .

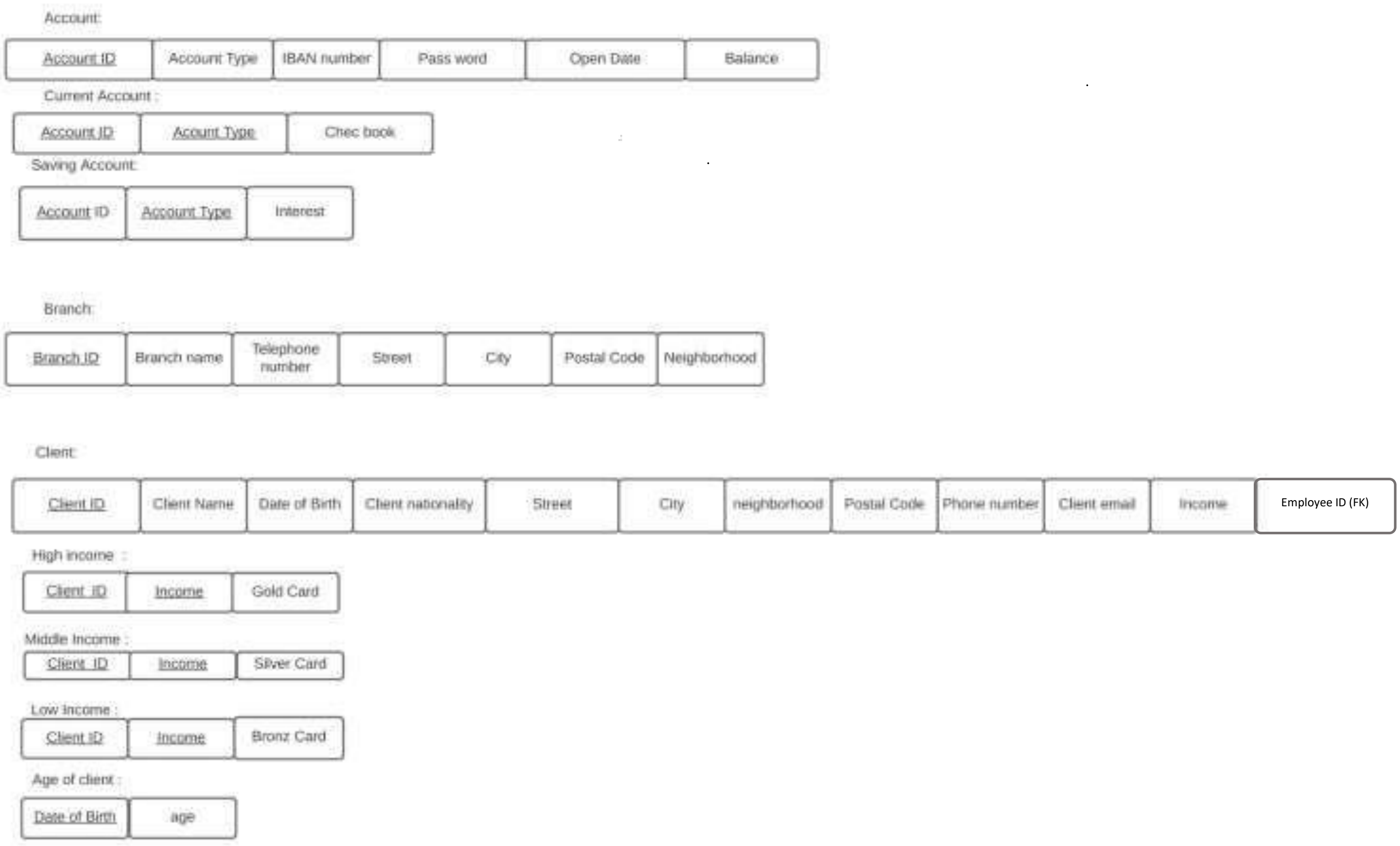
3- In The Client relation , it has a composite attribute (Client address) . So we have broken down the composite attribute into the following attributes: **city, street, neighborhood, postal code** , to become in 1NF . Then we checked it and did not find partial dependency but we find the transitive dependency between The date birth and Age , so we make a new relation To remove of the transitive dependency , we called it **Age of Client**. The date of birth in Client relation is FK from Age of Client relation . So it become in 3NF .
- High income relation , Middle income relation , Low income relation is a subtype from Client relation and the PK for all subtype is the Client ID and Income (FK from Client relation) . All of them in 3NF .

4- In The Loan relation , At first it was in 1NF . Then we checked it and did not find partial dependency but we find the transitive dependency between Loan Type attribute and Description , and between Total amount attribute and Remaining Amount . so we make a new relation we called it **Description of loan** and it PK is Loan Type . And we make anther new relation To remove of the transitive dependency between Total amount attribute and Remaining Amount , we called **the loan amount** and it PK is Total amount . Loan Type , Total amount in Loan relation is FK from Description of loan , the loan amount , Respectively . So it become in 3NF .
- Study Loan relation , Real Estate Loan relation is a subtype from Loan relation and the PK for all subtype is the Loan ID and Loan Type (FK from Loan relation) . All of them in 3NF .

- Student relation in 3NF . the student ID attribute in Study Loan relation is FK from Student relation .

5- In Account relation , At first it was in 1NF . Then we checked it and did not find partial dependency or transitive dependency so it is In 3NF . - Current account relation , Saving account relation is a subtype from Account relation and the PK for all subtype is the Account ID and Account Type (FK from Account relation) . All of them in 3NF .

2.2.2 : The final version of the ERD :



Account:

Account ID	Account Type	IBAN number	Pass word	Open Date	Balance

Current Account :

Account ID	Acount Type	Chec book

Saving Account:

Account ID	Account Type	Interest

Branch:

Branch ID	Branch name	Telephone number	Street	City	Postal Code	Neighborhood

Client:

Client ID	Client Name	Date of Birth	Client nationality	Street	City	neighborhood	Postal Code	Phone number	Client email	Income	Employee ID (FK)

High income :

Client ID	Income	Gold Card

Middle Income :

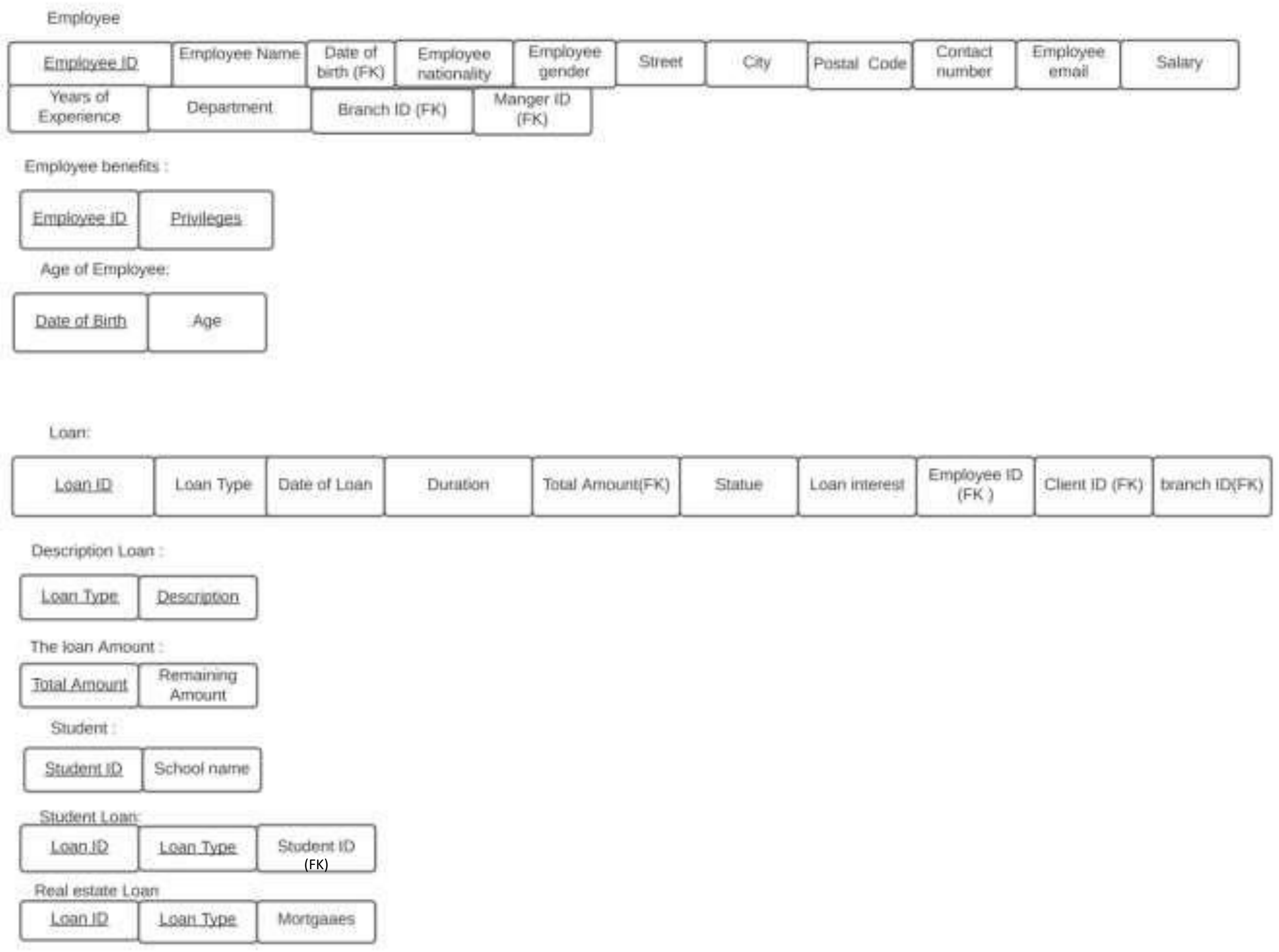
Client ID	Income	Silver Card

Low income :

Client ID	Income	Bronz Card

Age of client :

Date of Birth	age



Employee

Employee ID	Employee Name	Date of birth (FK)	Employee nationality	Employee gender	Street	City	Postal Code	Contact number	Employee email	Salary
Years of Experience	Department	Branch ID (FK)	Manger ID (FK)							

Employee benefits :

Employee ID	Privileges

Age of Employee:

Date of Birth	Age

Loan:

Loan ID	Loan Type	Date of Loan	Duration	Total Amount(FK)	Statue	Loan interest	Employee ID (FK)	Client ID (FK)	branch ID(FK)

Description Loan :

Loan Type	Description

The loan Amount :

Total Amount	Remaining Amount

Student :

Student ID	School name

Student Loan:

Loan ID	Loan Type	Student ID (FK)

Real estate Loan

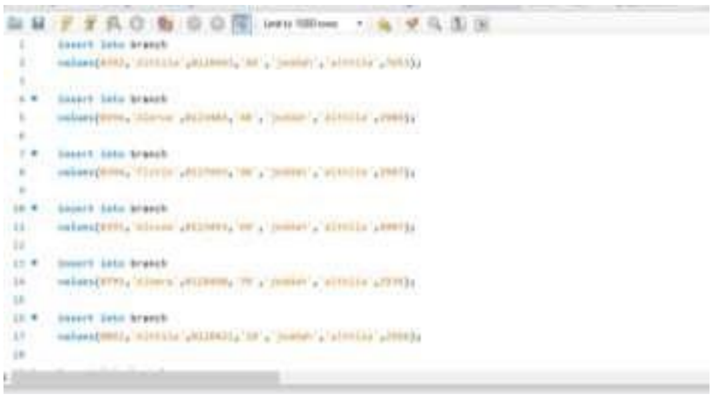
Loan ID	Loan Type	Mortgaaes

Chapter 3:DB Implementation

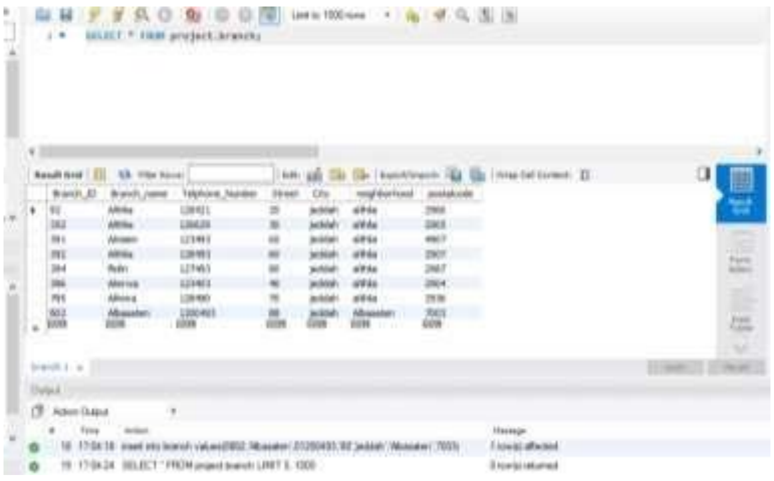
Team member	The Task
Ohod Tariq Alsheikh	The creation of: Loan table, Description loan table , the loan amount table , student table ,student loan table , real estate loan table . Insert query . Select query. JDBC Interface.
Raghad Fawzi Alsyed	The creation of: Branch table , client table , high income table , low income table , average income table , age of client table. Delete query. Update query. Order by query.
Nawal Awad Algrni	The creation of: Employee table ,Employee benefits table, age of employee , Account table , current account table , saving account table. Subquery. Group by query.

3.1 Create tables

Branch table



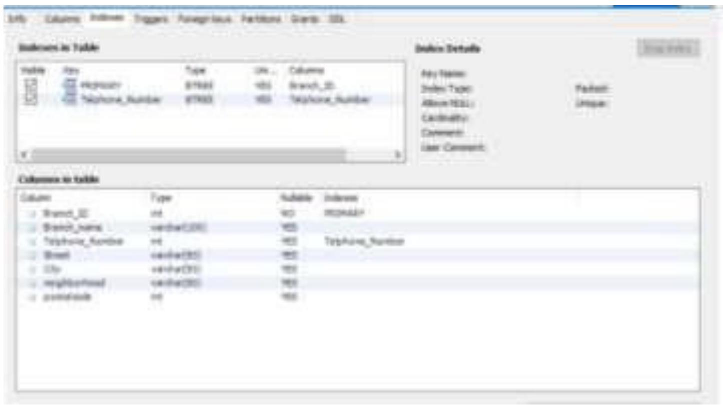
```
insert into branch
values([illegible]);

insert into branch
values([illegible]);

insert into branch
values([illegible]);

insert into branch
values([illegible]);

insert into branch
values([illegible]);

insert into branch
values([illegible]);
```



```
SELECT * FROM project.branch;
```

Branch_ID	Branch_name	Telephone_Number	Street	City	neighborhood	postalcode
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Jeddah	[illegible]	[illegible]
[illegible]	Albasaten	[illegible]	[illegible]	Jeddah	Albasaten	[illegible]
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
#   Time      Action                                                              Message
18  17:04:18  insert into branch values([illegible] 'Albasaten' [illegible])     1 row(s) affected
19  17:04:24  SELECT * FROM project.branch LIMIT 0, 1000                          8 row(s) returned
```



Indexes in Table

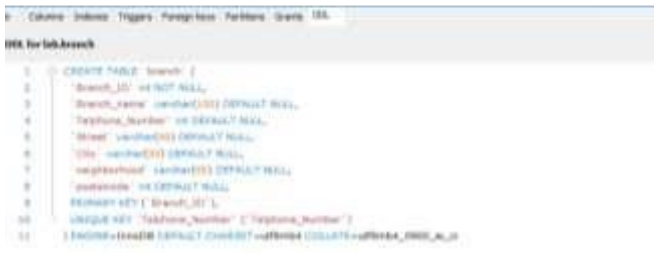
Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Branch_ID
✓	Telephone_Number	BTREE	YES	Telephone_Number

Index Details

Key Name:
Index Type:
Allow NULL:
Cardinality:
Comment:
User Comment:

Columns in table

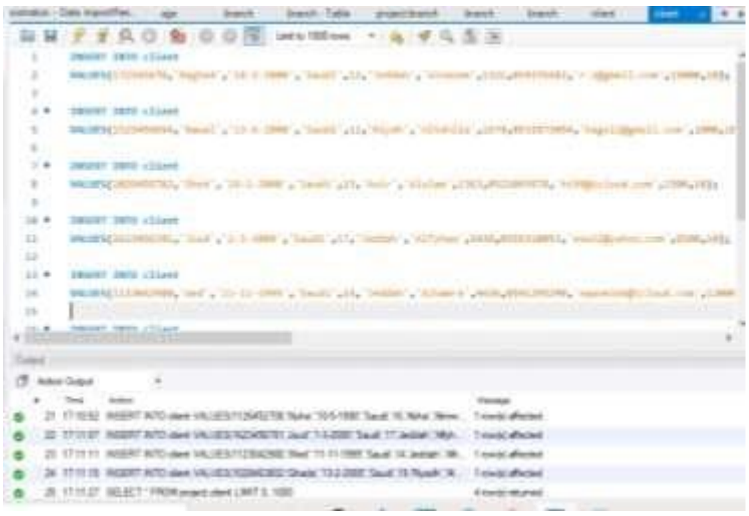
Column	Type	Nullable	Indexes
Branch_ID	int	NO	PRIMARY
Branch_name	varchar([illegible])	YES	
Telephone_Number	int	YES	Telephone_Number
Street	varchar([illegible])	YES	
City	varchar([illegible])	YES	
neighborhood	varchar([illegible])	YES	
postalcode	int	YES	



DDL for lab.branch

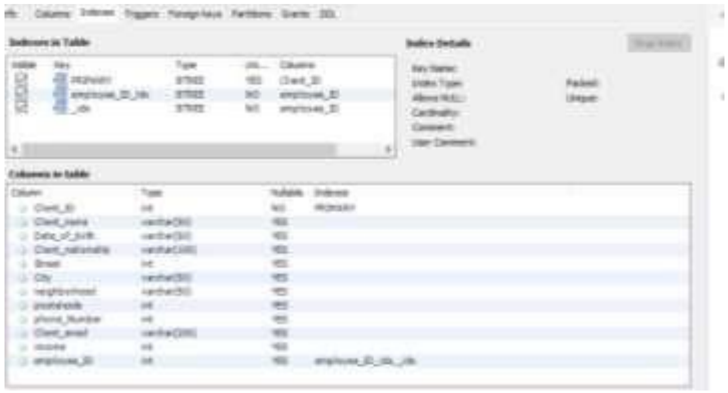
```
CREATE TABLE `branch` (
  `Branch_ID` int NOT NULL,
  `Branch_name` varchar(50) DEFAULT NULL,
  `Telphone_Number` int DEFAULT NULL,
  `Street` varchar(50) DEFAULT NULL,
  `City` varchar(50) DEFAULT NULL,
  `neighborhood` varchar(50) DEFAULT NULL,
  `postalcode` int DEFAULT NULL,
  PRIMARY KEY (`Branch_ID`),
  UNIQUE KEY `Telphone_Number` (`Telphone_Number`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Client table



```
INSERT INTO client
VALUES([illegible]
INSERT INTO client
VALUES([illegible]
INSERT INTO client
VALUES([illegible]
INSERT INTO client
VALUES([illegible]
INSERT INTO client
VALUES([illegible]

INSERT INTO client
```

```
#   Time      Action                                                                    Message
21  17:10:52  INSERT INTO client VALUES(1125452708,'Noha','10-5-1998','Saudi',15,'Noha','Nme...  1 row(s) affected
22  17:11:07  INSERT INTO client VALUES(1622456781,'Jaud','1-1-2000','Saudi',17,'Jeddah','Mlyh...  1 row(s) affected
23  17:11:11  INSERT INTO client VALUES(1123042568,'Wed','11-11-1995','Saudi',14,'Jeddah','Mi...  1 row(s) affected
24  17:11:15  INSERT INTO client VALUES(1022463802,'Ghada','13-2-2000','Saudi',15,'Riyadh','A...  1 row(s) affected
25  17:11:27  SELECT * FROM project.client LIMIT 0, 1000                                          4 row(s) returned
```


Info Columns Indexes Triggers Foreign keys Partitions Grants DDL
Drop Index

Indexes in Table

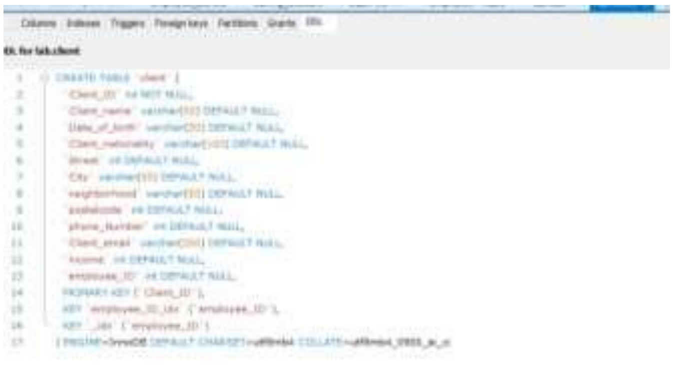
Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Client_ID
✓	employee_ID_idx	BTREE	NO	employee_ID
✓	_idx	BTREE	NO	employee_ID

Index Details

Key Name:	
Index Type:	Packed:
Allows NULL:	Unique:
Cardinality:	
Comment:	
User Comment:	

Columns in table

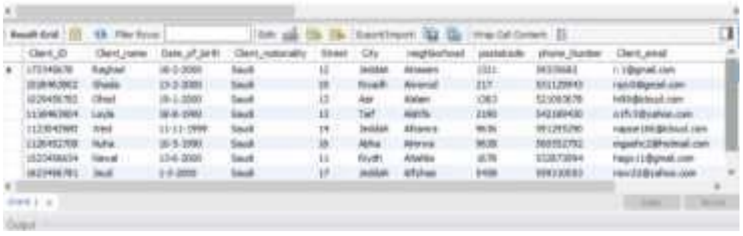
Column	Type	Nullable	Indexes
Client_ID	int	NO	PRIMARY
Client_name	varchar(50)	YES	
Date_of_birth	varchar(50)	YES	
Client_nationality	varchar(100)	YES	
Street	int	YES	
City	varchar(50)	YES	
neighborhood	varchar(50)	YES	
postalcode	int	YES	
phone_Number	int	YES	
Client_email	varchar(100)	YES	
income	int	YES	
employee_ID	int	YES	employee_ID_idx, _idx



Columns Indexes Triggers Foreign keys Partitions Grants DDL

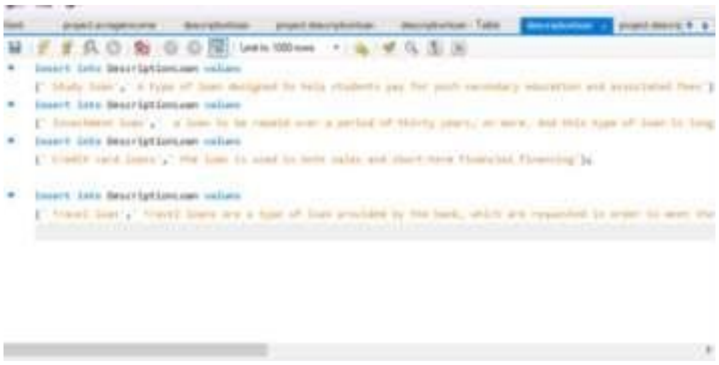
DDL for lab.client

```
CREATE TABLE `client` (
  `Client_ID` int NOT NULL,
  `Client_name` varchar([illegible]) DEFAULT NULL,
  `Date_of_birth` varchar([illegible]) DEFAULT NULL,
  `Client_nationality` varchar([illegible]) DEFAULT NULL,
  `Street` int DEFAULT NULL,
  `City` varchar([illegible]) DEFAULT NULL,
  `neighborhood` varchar([illegible]) DEFAULT NULL,
  `postalcode` int DEFAULT NULL,
  `phone_Number` int DEFAULT NULL,
  `Client_email` varchar([illegible]) DEFAULT NULL,
  `Income` int DEFAULT NULL,
  `employee_ID` int DEFAULT NULL,
  PRIMARY KEY (`Client_ID`),
  KEY `employee_ID_idx` (`employee_ID`),
  KEY `_idx` (`employee_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

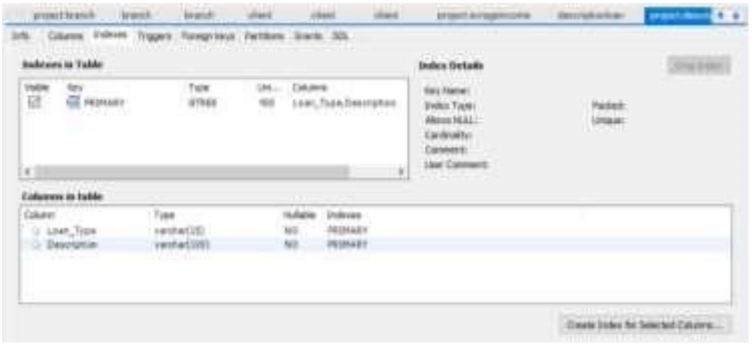


Client_ID	Client_name	Date_of_birth	Client_nationality	Street	City	neighborhood	postalcode	phone_Number	Client_email
[illegible]	Raghad	16-2-2000	Saudi	12	Jeddah	Alnaeem	[illegible]	[illegible]	[illegible]@gmail.com
[illegible]	Shada	13-2-2000	Saudi	18	Riyadh	Alnernal	217	[illegible]	[illegible]@gmail.com
[illegible]	Ohad	18-1-2000	Saudi	13	Asir	Alalam	[illegible]	[illegible]	[illegible]@icloud.com
[illegible]	Layla	18-8-1999	Saudi	13	Taif	Alattfa	2190	[illegible]	[illegible]@yahoo.com
[illegible]	Ared	11-11-1999	Saudi	14	Jeddah	Alhamra	[illegible]	[illegible]	[illegible]@icloud.com
[illegible]	Nuha	16-5-1999	Saudi	18	Abha	Almrwa	[illegible]	[illegible]	[illegible]@hotmail.com
[illegible]	Nawal	13-6-2000	Saudi	11	Riyadh	Alsahba	[illegible]	[illegible]	[illegible]@gmail.com
[illegible]	Saud	1-7-2000	Saudi	17	Jeddah	Alfyhaa	[illegible]	[illegible]	[illegible]@yahoo.com

Types of Loans tables



```
Insert into DescriptionLoan values
(' Study loan ', ' A type of loan designed to help students pay for post-secondary education and associated fees' )
Insert into DescriptionLoan values
(' Investment loan ', ' a loan to be repaid over a period of thirty years, or more, and this type of loan is long
Insert into DescriptionLoan values
(' Credit card loans ',' The loan is used to both sales and short-term financial financing' );

Insert into DescriptionLoan values
(' Travel loan ', ' Travel loans are a type of loan provided by the bank, which are requested in order to meet the
```



Info Columns Indexes Triggers Foreign keys Partitions Grants SQL

Indexes in Table

Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Loan_Type,Description

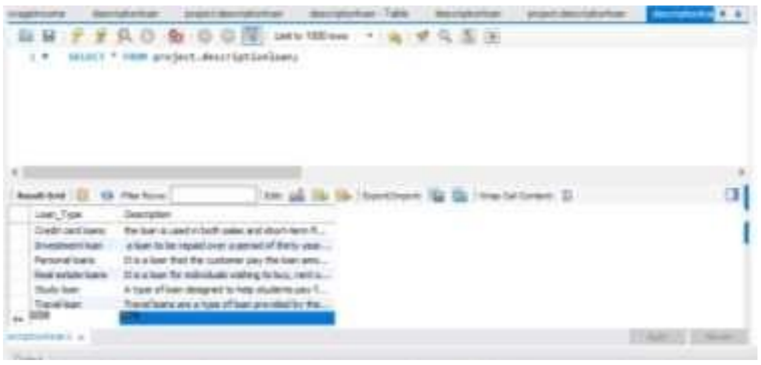
Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

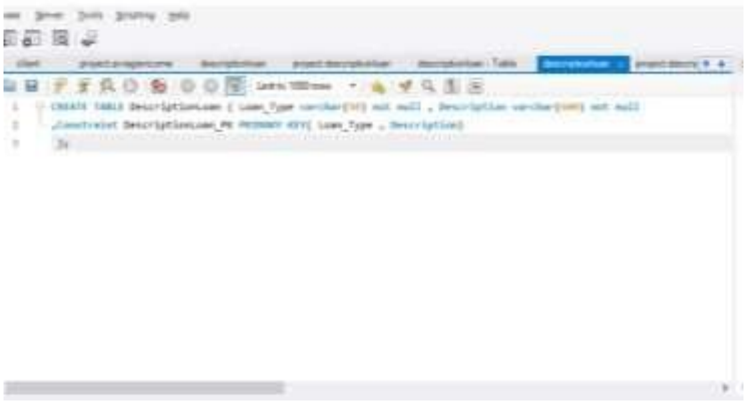
Column	Type	Nullable	Indexes
Loan_Type	varchar(20)	NO	PRIMARY
Description	varchar(300)	NO	PRIMARY

Create Index for Selected Columns...



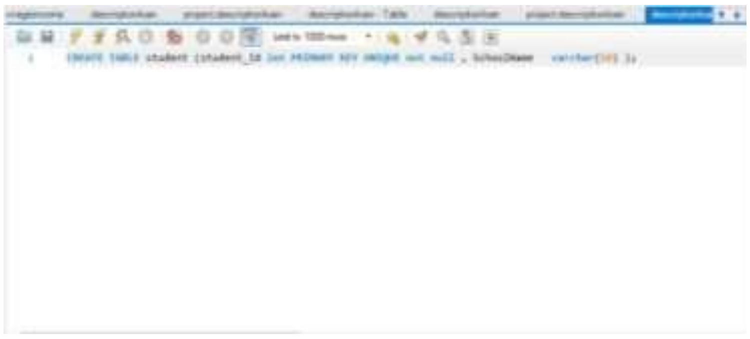
```
SELECT * FROM project.descriptionloan;
```

Loan_Type	Description
Credit card loans	the loan is used in both sales and short-term fi...
Investment loan	a loan to be repaid over a period of thirty year...
Personal loans	It is a loan that the customer pay the loan amo...
Real estate loans	It is a loan for individuals wishing to buy, rent o...
Study loan	A type of loan designed to help students pay f...
Travel loan	Travel loans are a type of loan provided by the...
NULL	NULL

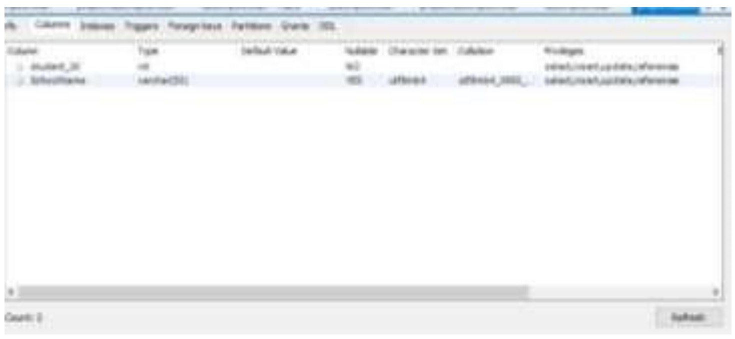


```
CREATE TABLE DescriptionLoan ( Loan_Type varchar(30) not null , Description varchar(500) not null
,Constraint DescriptionLoan_PK PRIMARY KEY( Loan_Type , Description)
);
```

Student tables

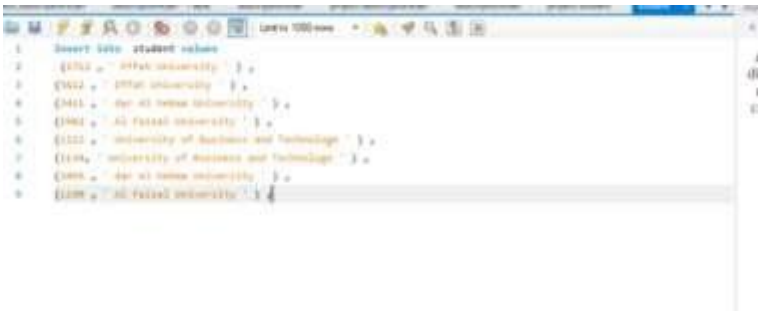


```
CREATE TABLE student (student_Id int PRIMARY KEY UNIQUE not null , SchoolName  varchar(50) );
```

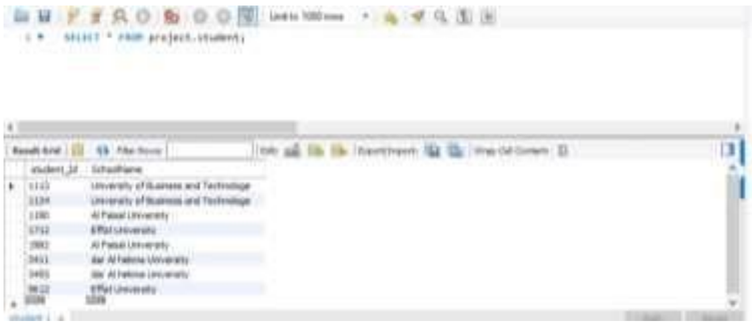


Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
student_Id	int		NO			select,insert,update,references
SchoolName	varchar(50)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Count: 2



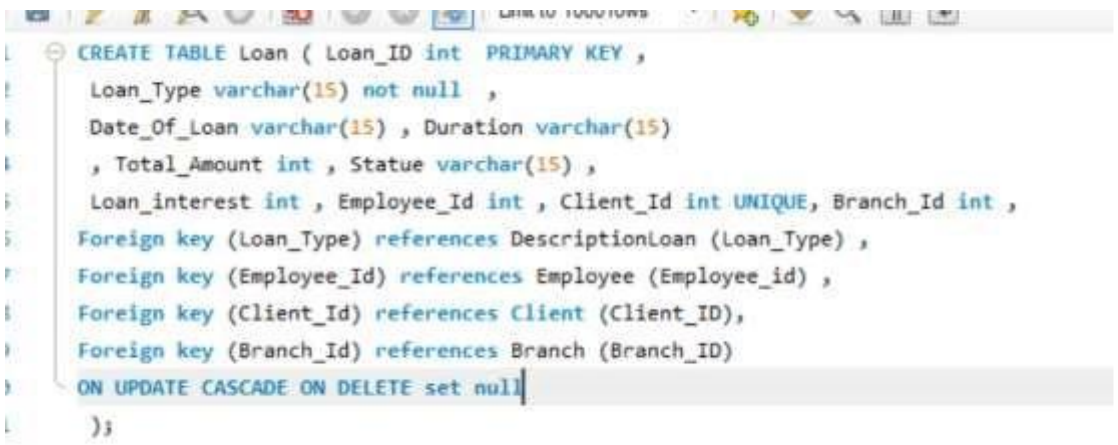
```
Insert into student values
(1712 , ' Effat University ' ) ,
(9612 , ' Effat University ' ) ,
(3411 , ' dar Al hekma University ' ) ,
(2982 , ' Al Faisal University ' ) ,
(1113 , ' University of Business and Technology ' ) ,
(1134, ' University of Business and Technology ' ) ,
(3401 , ' dar Al hekma University ' ) ,
(1180 , ' Al Faisal University ' ) ;
```



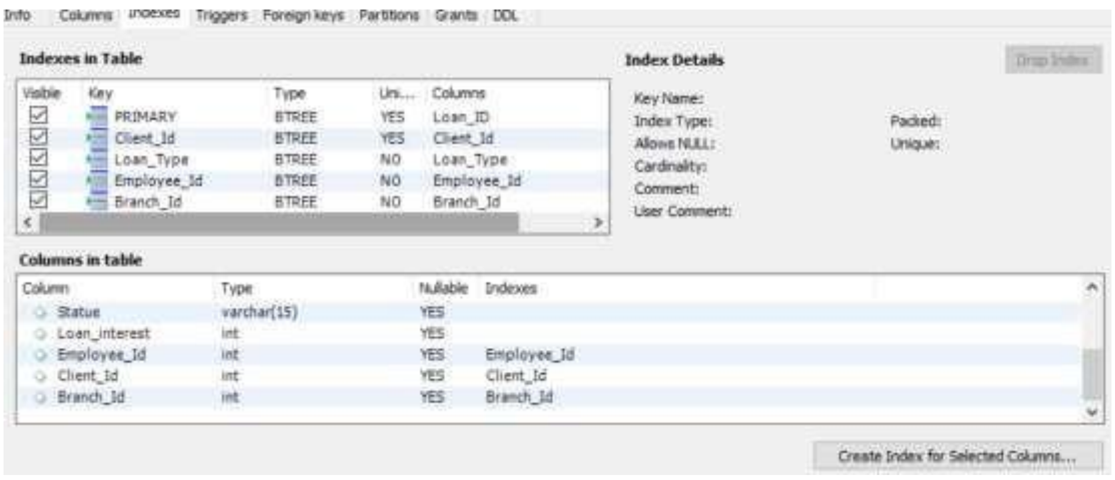
```
SELECT * FROM project.student;
```

student_Id	SchoolName
1113	University of Business and Technology
1134	University of Business and Technology
1180	Al Faisal University
1712	Effat University
2982	Al Faisal University
3411	dar Al hekma University
3401	dar Al hekma University
9612	Effat University
NULL	NULL

Loans tables



```
CREATE TABLE Loan ( Loan_ID int  PRIMARY KEY ,
 Loan_Type varchar(15) not null  ,
 Date_Of_Loan varchar(15) , Duration varchar(15)
 , Total_Amount int , Statue varchar(15) ,
 Loan_interest int , Employee_Id int , Client_Id int UNIQUE, Branch_Id int ,
Foreign key (Loan_Type) references DescriptionLoan (Loan_Type) ,
Foreign key (Employee_Id) references Employee (Employee_id) ,
Foreign key (Client_Id) references Client (Client_ID),
Foreign key (Branch_Id) references Branch (Branch_ID)
ON UPDATE CASCADE ON DELETE set null
 );
```



Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	Loan_ID
☑	Client_Id	BTREE	YES	Client_Id
☑	Loan_Type	BTREE	NO	Loan_Type
☑	Employee_Id	BTREE	NO	Employee_Id
☑	Branch_Id	BTREE	NO	Branch_Id

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

Column	Type	Nullable	Indexes
Statue	varchar(15)	YES	
Loan_interest	int	YES	
Employee_Id	int	YES	Employee_Id
Client_Id	int	YES	Client_Id
Branch_Id	int	YES	Branch_Id

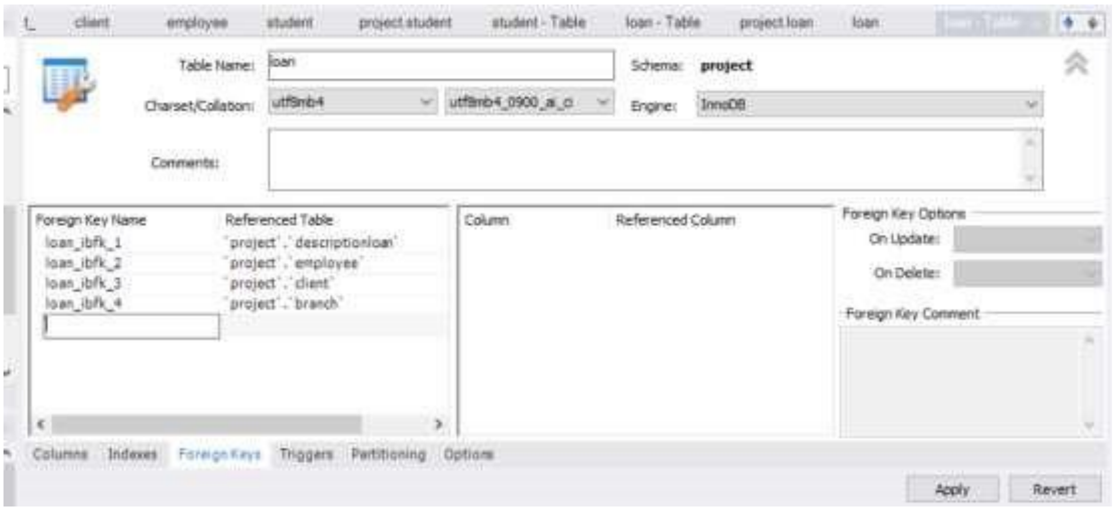
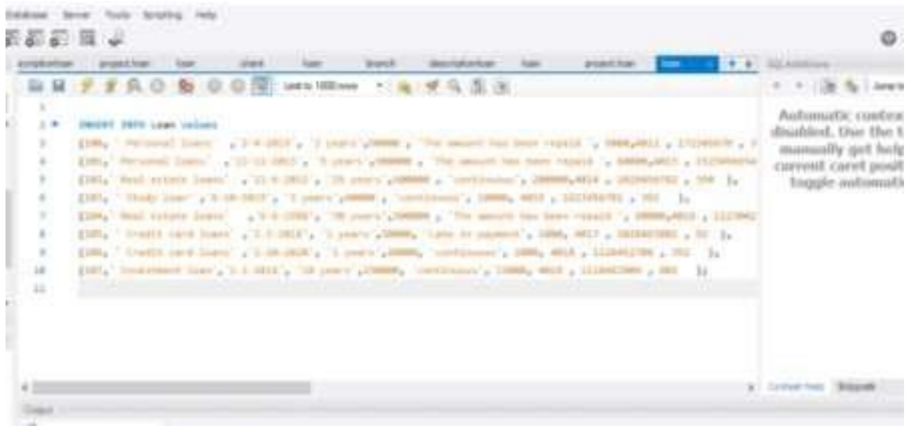


Table Name: loan
Schema: project
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

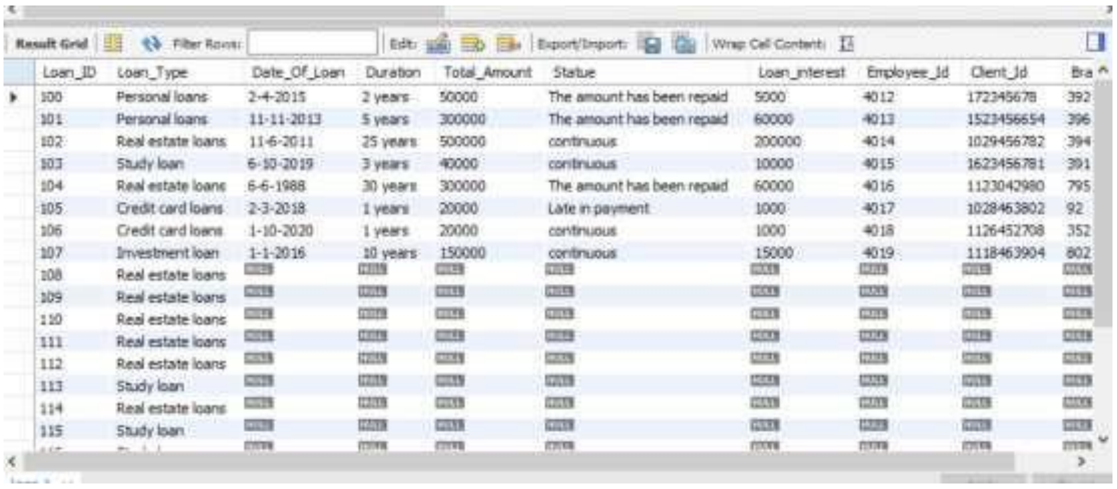
Foreign Key Name	Referenced Table
loan_ibfk_1	`project`.`descriptionloan`
loan_ibfk_2	`project`.`employee`
loan_ibfk_3	`project`.`client`
loan_ibfk_4	`project`.`branch`

Column	Referenced Column

Foreign Key Options
On Update:
On Delete:
Foreign Key Comment


Database
Server
Tools
Scripting
Help
Limit to 1000 rows
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

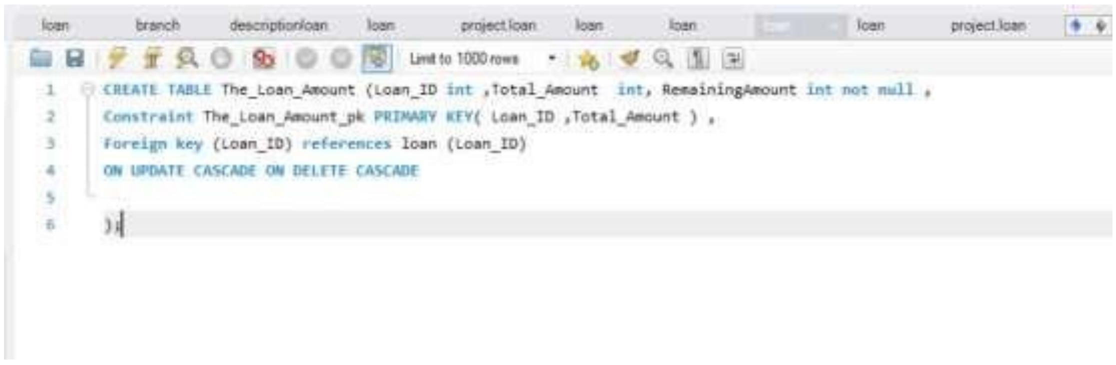
```
INSERT INTO Loan values
(100, ' Personal loans' , '1-4-2019', '2 years',20000 , 'The amount has been repaid ', [illegible], 4011 , [illegible] , [illegible]
(101, ' Personal loans' , '11-11-2015' , '5 years',[illegible] , 'The amount has been repaid ' , [illegible], 4015 , [illegible]
(102, ' Real estate loans' , '11-9-2012' , '25 years',[illegible] , 'continuous' , [illegible], 4014 , [illegible] , [illegible] ),
(103, ' Study loan' , '3-10-2017', '3 years',[illegible] , 'continuous' , [illegible], 4010 , [illegible] , [illegible] ),
(104, ' Real estate loans' , '5-6-2008', '20 years',[illegible] , 'The amount has been repaid ' , [illegible], 4013 , [illegible]
(105, ' Credit card loans' , '2-7-2018', '1 years',[illegible], 'Late in payment' , [illegible], 4017 , [illegible] , [illegible] ),
(106, ' Credit card loans' , '1-10-2020', '1 years',[illegible], 'continuous' , [illegible], 4018 , [illegible] , [illegible] ),
(107, ' Investment loan', '1-1-2013', '10 years',[illegible], 'continuous' , [illegible], 4012 , [illegible] , [illegible] );
```



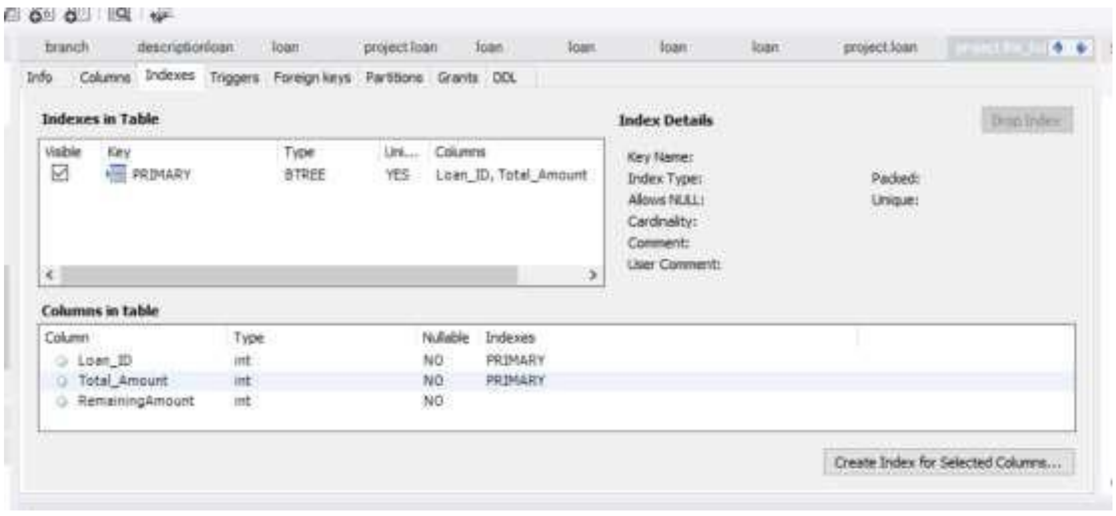
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Loan_ID	Loan_Type	Date_Of_Loan	Duration	Total_Amount	Statue	Loan_interest	Employee_Id	Client_Id	Bra
100	Personal loans	2-4-2015	2 years	50000	The amount has been repaid	5000	4012	172345678	392
101	Personal loans	11-11-2013	5 years	300000	The amount has been repaid	60000	4013	1523456654	396
102	Real estate loans	11-6-2011	25 years	500000	continuous	200000	4014	1029456782	394
103	Study loan	6-10-2019	3 years	40000	continuous	10000	4015	1623456781	391
104	Real estate loans	6-6-1988	30 years	300000	The amount has been repaid	60000	4016	1123042980	795
105	Credit card loans	2-3-2018	1 years	20000	Late in payment	1000	4017	1028463802	92
106	Credit card loans	1-10-2020	1 years	20000	continuous	1000	4018	1126452708	352
107	Investment loan	1-1-2016	10 years	150000	continuous	15000	4019	1118463904	802
108	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
109	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
110	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
111	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
112	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
113	Study loan	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
114	Real estate loans	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
115	Study loan	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Amount of loan table



```sql
CREATE TABLE The_Loan_Amount (Loan_ID int ,Total_Amount  int, RemainingAmount int not null ,
Constraint The_Loan_Amount_pk PRIMARY KEY( Loan_ID ,Total_Amount ) ,
Foreign key (Loan_ID) references loan (Loan_ID)
ON UPDATE CASCADE ON DELETE CASCADE

);
```



Indexes in Table

Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Loan_ID, Total_Amount

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

Column	Type	Nullable	Indexes
Loan_ID	int	NO	PRIMARY
Total_Amount	int	NO	PRIMARY
RemainingAmount	int	NO	

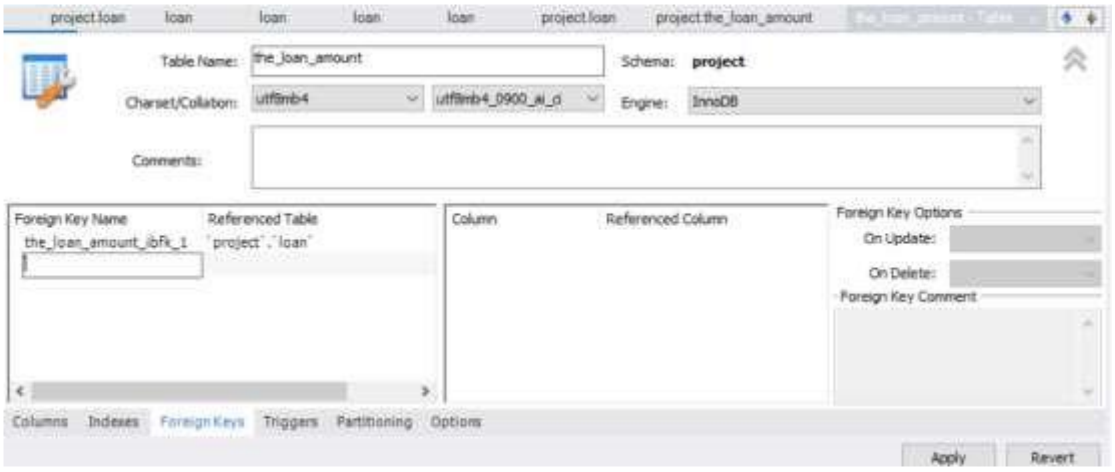
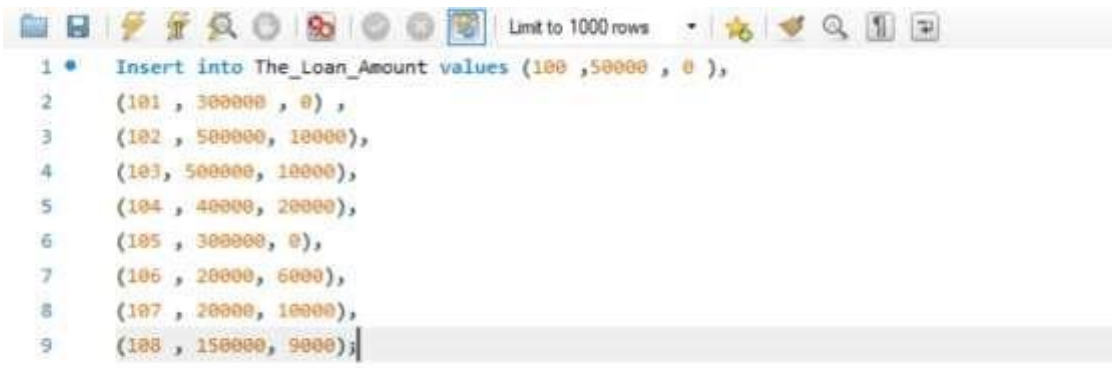


Table Name: the_loan_amount
Schema: project
Charset/Collation: utf8mb4 | utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

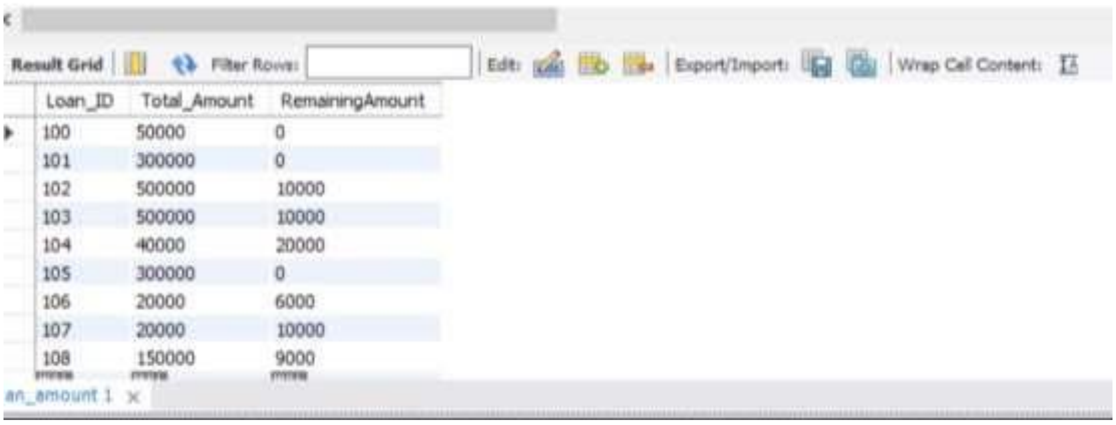
Foreign Key Name	Referenced Table
the_loan_amount_ibfk_1	`project`.`loan`

Column | Referenced Column

Foreign Key Options
On Update:
On Delete:
Foreign Key Comment

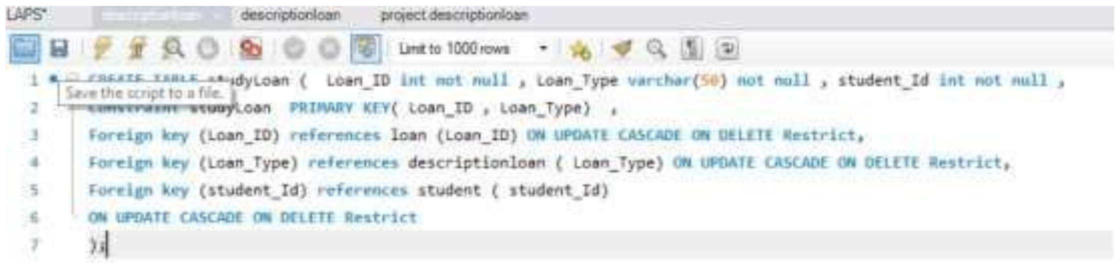


```
Insert into The_Loan_Amount values (100 ,50000 , 0 ),
(101 , 300000 , 0) ,
(102 , 500000, 10000),
(103, 500000, 10000),
(104 , 40000, 20000),
(105 , 300000, 0),
(106 , 20000, 6000),
(107 , 20000, 10000),
(108 , 150000, 9000);
```

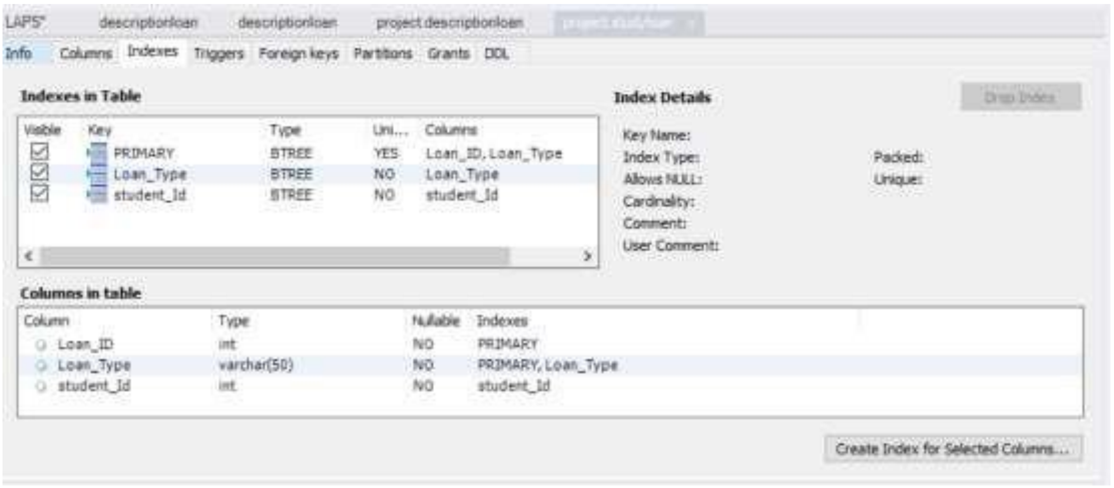


Loan_ID	Total_Amount	RemainingAmount
100	50000	0
101	300000	0
102	500000	10000
103	500000	10000
104	40000	20000
105	300000	0
106	20000	6000
107	20000	10000
108	150000	9000

Study loan table



```
CREATE TABLE studyLoan (  Loan_ID int not null , Loan_Type varchar(50) not null , student_Id int not null ,
constraint studyLoan  PRIMARY KEY( Loan_ID , Loan_Type)  ,
Foreign key (Loan_ID) references loan (Loan_ID) ON UPDATE CASCADE ON DELETE Restrict,
Foreign key (Loan_Type) references descriptionloan ( Loan_Type) ON UPDATE CASCADE ON DELETE Restrict,
Foreign key (student_Id) references student ( student_Id)
ON UPDATE CASCADE ON DELETE Restrict
);
```



Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	Loan_ID, Loan_Type
☑	Loan_Type	BTREE	NO	Loan_Type
☑	student_Id	BTREE	NO	student_Id

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

Column	Type	Nullable	Indexes
Loan_ID	int	NO	PRIMARY
Loan_Type	varchar(50)	NO	PRIMARY, Loan_Type
student_Id	int	NO	student_Id

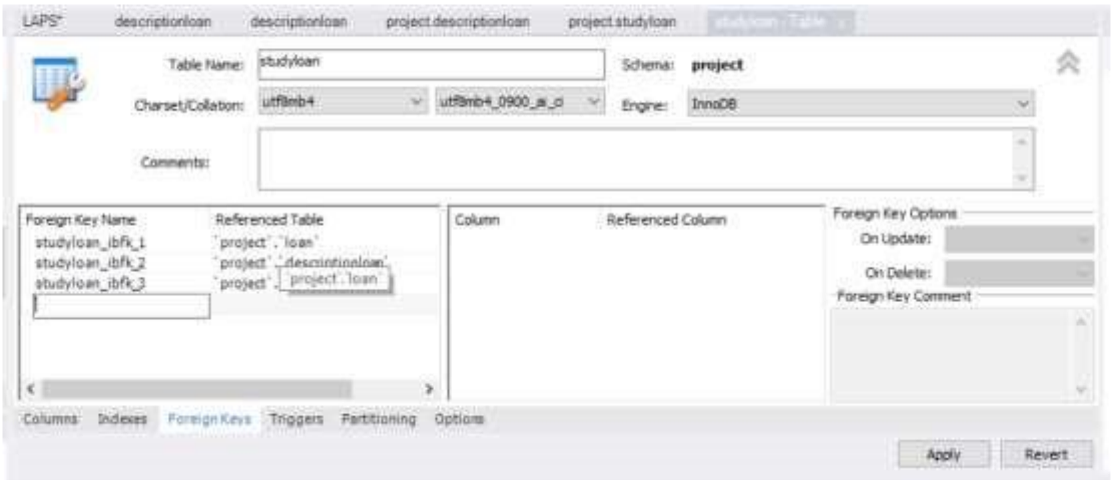
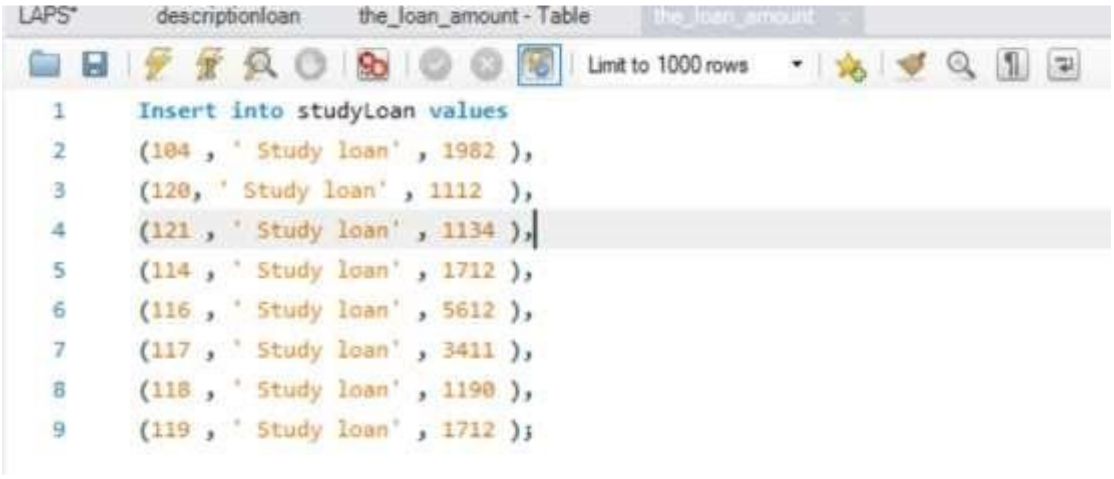
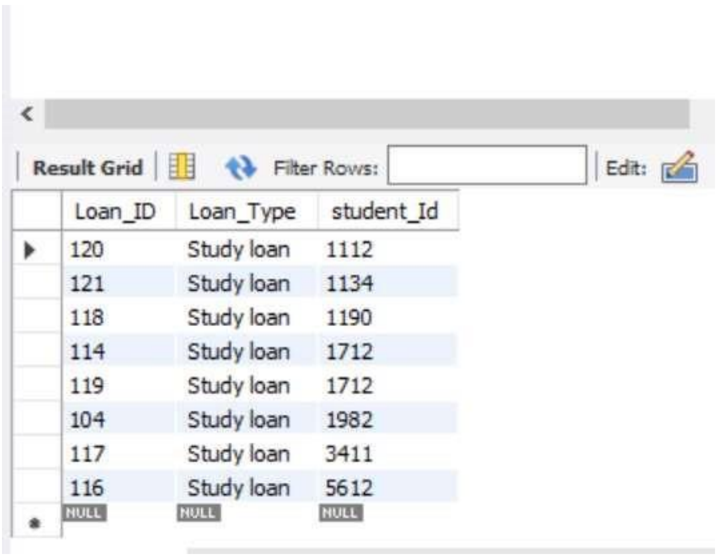


Table Name: studyloan
Schema: project
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

Foreign Key Name	Referenced Table
studyloan_ibfk_1	`project`.`loan`
studyloan_ibfk_2	`project`.`descriptionloan`
studyloan_ibfk_3	`project`.`loan`

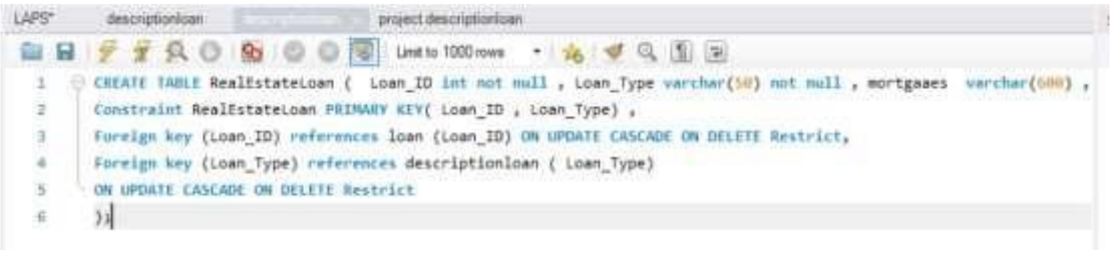


```
Insert into studyLoan values
(104 , ' Study loan' , 1982 ),
(120, ' Study loan' , 1112  ),
(121 , ' Study loan' , 1134 ),
(114 , ' Study loan' , 1712 ),
(116 , ' Study loan' , 5612 ),
(117 , ' Study loan' , 3411 ),
(118 , ' Study loan' , 1190 ),
(119 , ' Study loan' , 1712 );
```

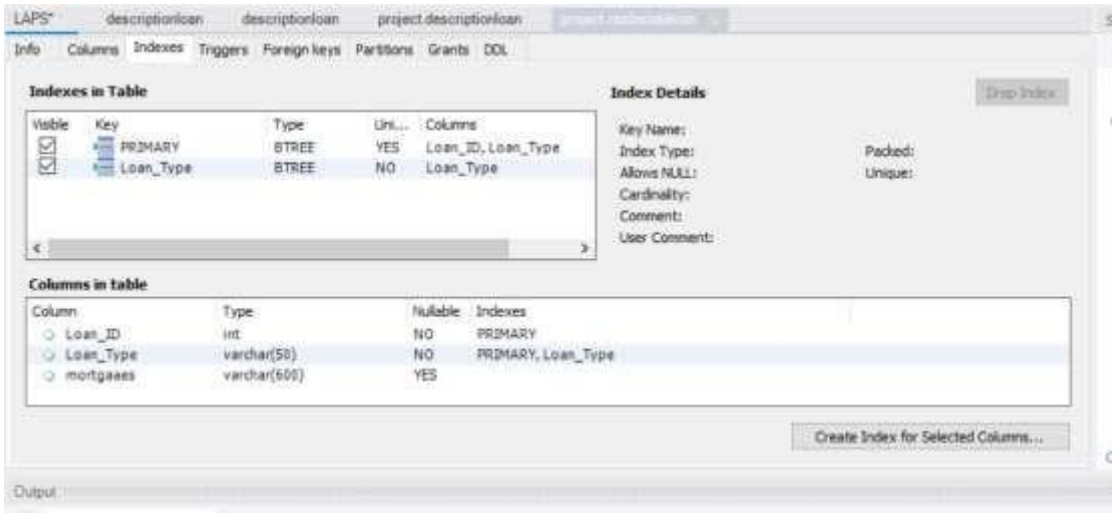


Loan_ID	Loan_Type	student_Id
120	Study loan	1112
121	Study loan	1134
118	Study loan	1190
114	Study loan	1712
119	Study loan	1712
104	Study loan	1982
117	Study loan	3411
116	Study loan	5612
NULL	NULL	NULL

Real estate table



```
CREATE TABLE RealEstateLoan (  Loan_ID int not null , Loan_Type varchar(50) not null , mortgaaes  varchar(600) ,
Constraint RealEstateLoan PRIMARY KEY( Loan_ID , Loan_Type) ,
Foreign key (Loan_ID) references loan (Loan_ID) ON UPDATE CASCADE ON DELETE Restrict,
Foreign key (Loan_Type) references descriptionloan ( Loan_Type)
ON UPDATE CASCADE ON DELETE Restrict
);
```



Indexes in Table

Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Loan_ID, Loan_Type
✓	Loan_Type	BTREE	NO	Loan_Type

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

Column	Type	Nullable	Indexes
Loan_ID	int	NO	PRIMARY
Loan_Type	varchar(50)	NO	PRIMARY, Loan_Type
mortgaaes	varchar(600)	YES	

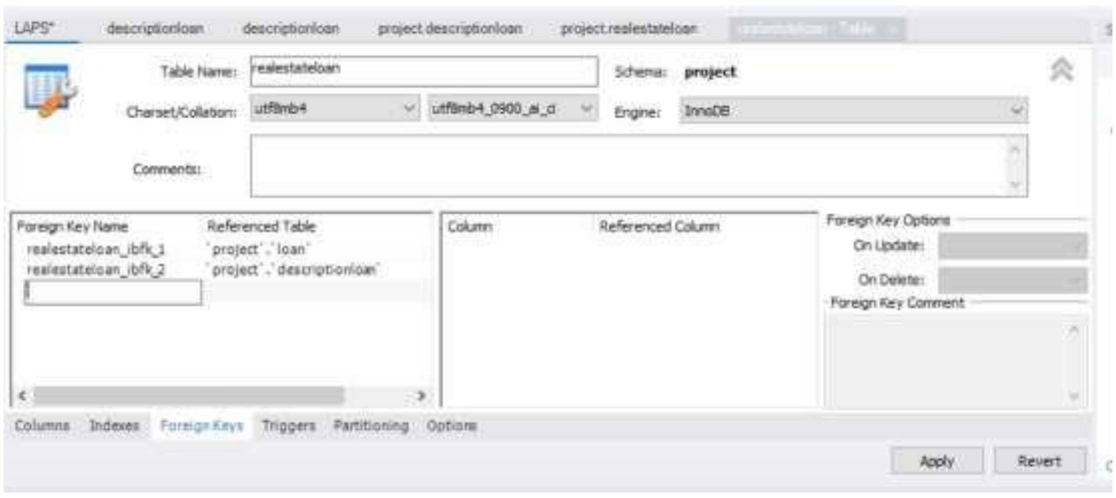
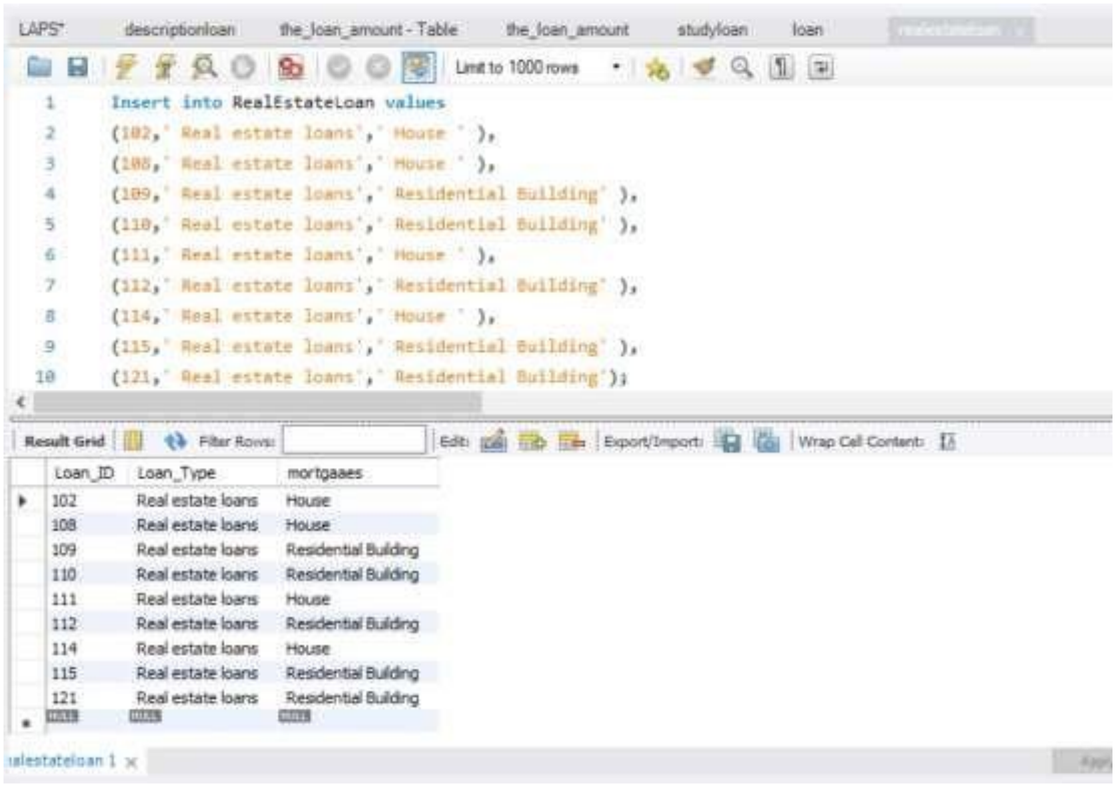


Table Name: realestateloan
Schema: project
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

Foreign Key Name	Referenced Table
realestateloan_ibfk_1	`project`.`loan`
realestateloan_ibfk_2	`project`.`descriptionloan`

Column	Referenced Column

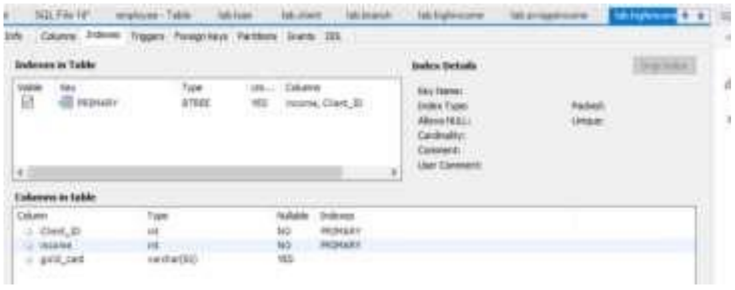
Foreign Key Options
On Update:
On Delete:
Foreign Key Comment



```
Insert into RealEstateLoan values
(102,' Real estate loans',' House ' ),
(108,' Real estate loans',' House ' ),
(109,' Real estate loans',' Residential Building' ),
(110,' Real estate loans',' Residential Building' ),
(111,' Real estate loans',' House ' ),
(112,' Real estate loans',' Residential Building' ),
(114,' Real estate loans',' House ' ),
(115,' Real estate loans',' Residential Building' ),
(121,' Real estate loans',' Residential Building');
```

Loan_ID	Loan_Type	mortgaaes
102	Real estate loans	House
108	Real estate loans	House
109	Real estate loans	Residential Building
110	Real estate loans	Residential Building
111	Real estate loans	House
112	Real estate loans	Residential Building
114	Real estate loans	House
115	Real estate loans	Residential Building
121	Real estate loans	Residential Building
NULL	NULL	NULL

Types of incomes



Indexes in Table

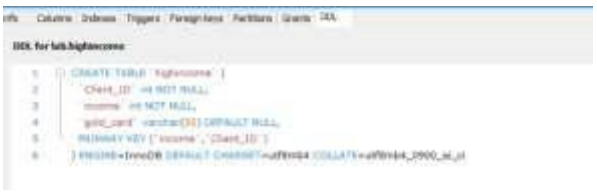
Visible	Key	Type	Uni...	Columns
	PRIMARY	BTREE	YES	income, Client_ID

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

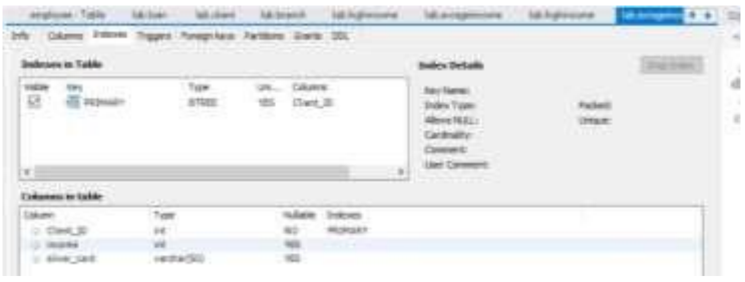
Columns in table

Column	Type	Nullable	Indexes
Client_ID	int	NO	PRIMARY
income	int	NO	PRIMARY
gold_card	varchar(50)	YES	



DDL for lab.highincome

```
CREATE TABLE `highincome` (
  `Client_ID` int NOT NULL,
  `income` int NOT NULL,
  `gold_card` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`income`,`Client_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```



Indexes in Table

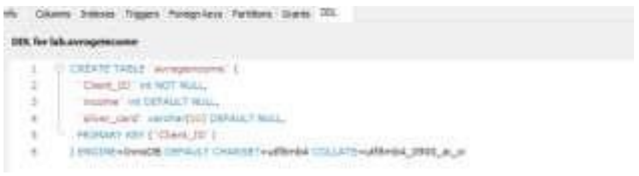
Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Client_ID

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

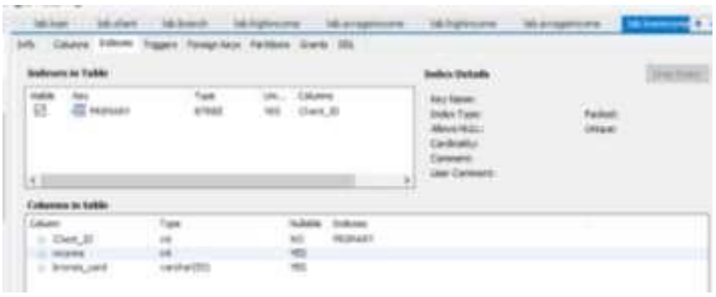
Columns in table

Column	Type	Nullable	Indexes
Client_ID	int	NO	PRIMARY
income	int	YES	
silver_card	varchar(50)	YES	



DDL for lab.averageincome

```
CREATE TABLE `averageincome` (
  `Client_ID` int NOT NULL,
  `income` int DEFAULT NULL,
  `silver_card` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Client_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```



Indexes in Table

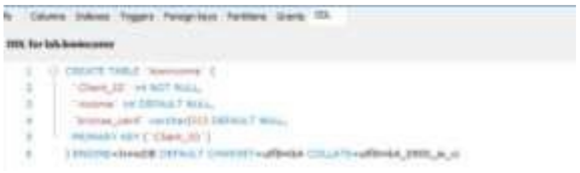
Visible	Key	Type	Uni...	Columns
✓	PRIMARY	BTREE	YES	Client_ID

Index Details

Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:
Packed:
Unique:

Columns in table

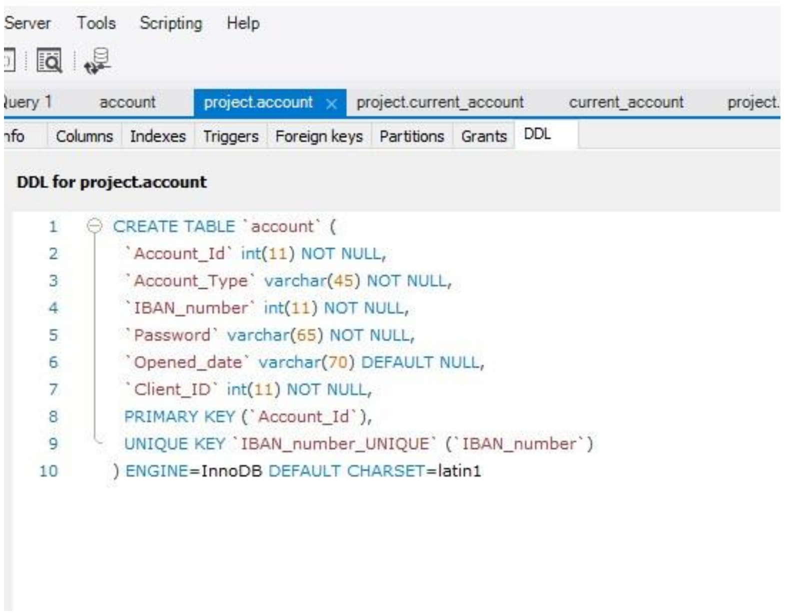
Column	Type	Nullable	Indexes
Client_ID	int	NO	PRIMARY
income	int	YES	
bronze_card	varchar(50)	YES	



DDL for lab.lowincome

```
CREATE TABLE `lowincome` (
  `Client_ID` int NOT NULL,
  `income` int DEFAULT NULL,
  `bronze_card` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Client_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

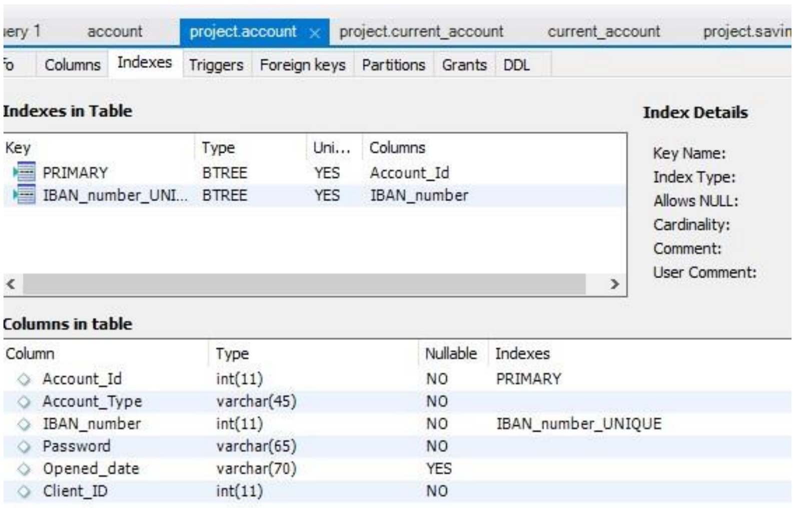
Account Table



Server Tools Scripting Help
Query 1 account project.account project.current_account current_account project.
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL

DDL for project.account

```
CREATE TABLE `account` (
  `Account_Id` int(11) NOT NULL,
  `Account_Type` varchar(45) NOT NULL,
  `IBAN_number` int(11) NOT NULL,
  `Password` varchar(65) NOT NULL,
  `Opened_date` varchar(70) DEFAULT NULL,
  `Client_ID` int(11) NOT NULL,
  PRIMARY KEY (`Account_Id`),
  UNIQUE KEY `IBAN_number_UNIQUE` (`IBAN_number`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```



Indexes in Table

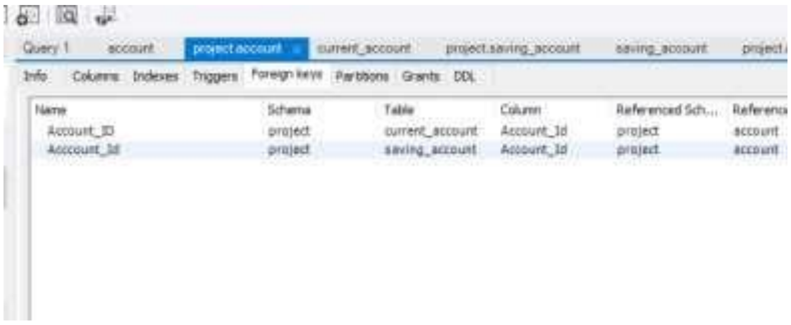
Key	Type	Uni...	Columns
PRIMARY	BTREE	YES	Account_Id
IBAN_number_UNI...	BTREE	YES	IBAN_number

Index Details

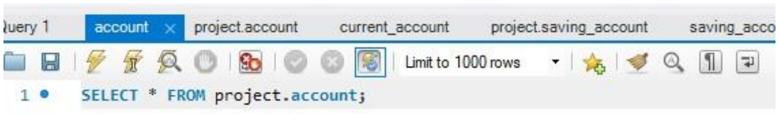
Key Name:
Index Type:
Allows NULL:
Cardinality:
Comment:
User Comment:

Columns in table

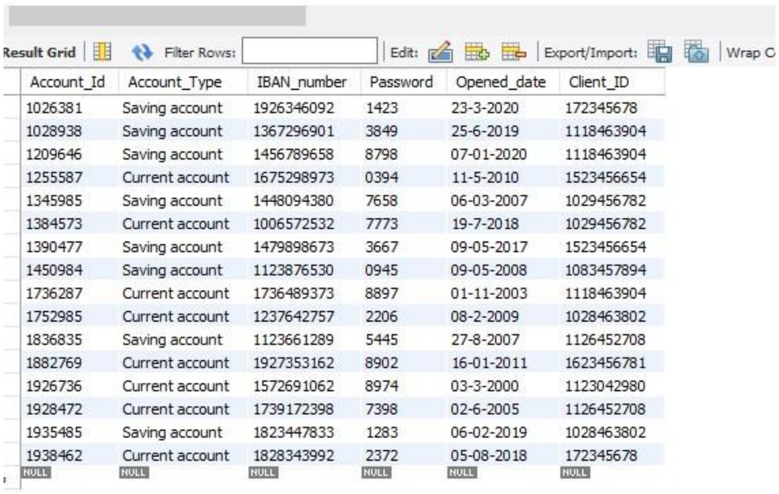
Column	Type	Nullable	Indexes
Account_Id	int(11)	NO	PRIMARY
Account_Type	varchar(45)	NO	
IBAN_number	int(11)	NO	IBAN_number_UNIQUE
Password	varchar(65)	NO	
Opened_date	varchar(70)	YES	
Client_ID	int(11)	NO	



Name	Schema	Table	Column	Referenced Sch...	Referenc
Account_ID	project	current_account	Account_Id	project	account
Account_Id	project	saving_account	Account_Id	project	account

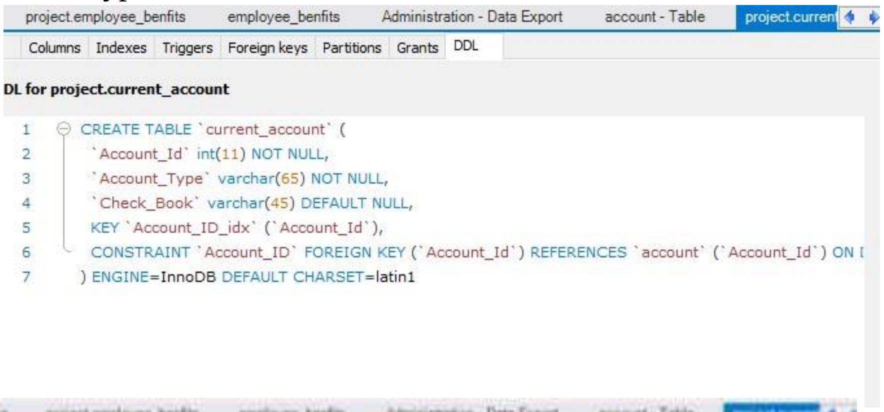


```
SELECT * FROM project.account;
```



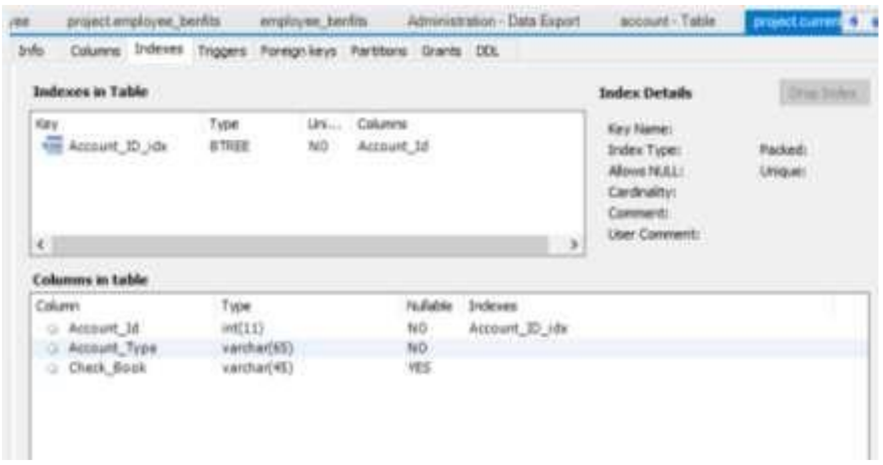
Account_Id	Account_Type	IBAN_number	Password	Opened_date	Client_ID
1026381	Saving account	1926346092	1423	23-3-2020	172345678
1028938	Saving account	1367296901	3849	25-6-2019	1118463904
1209646	Saving account	1456789658	8798	07-01-2020	1118463904
1255587	Current account	1675298973	0394	11-5-2010	1523456654
1345985	Saving account	1448094380	7658	06-03-2007	1029456782
1384573	Current account	1006572532	7773	19-7-2018	1029456782
1390477	Saving account	1479898673	3667	09-05-2017	1523456654
1450984	Saving account	1123876530	0945	09-05-2008	1083457894
1736287	Current account	1736489373	8897	01-11-2003	1118463904
1752985	Current account	1237642757	2206	08-2-2009	1028463802
1836835	Saving account	1123661289	5445	27-8-2007	1126452708
1882769	Current account	1927353162	8902	16-01-2011	1623456781
1926736	Current account	1572691062	8974	03-3-2000	1123042980
1928472	Current account	1739172398	7398	02-6-2005	1126452708
1935485	Saving account	1823447833	1283	06-02-2019	1028463802
1938462	Current account	1828343992	2372	05-08-2018	172345678
NULL	NULL	NULL	NULL	NULL	NULL

Account Types



DL for project.current_account

```
CREATE TABLE `current_account` (
  `Account_Id` int(11) NOT NULL,
  `Account_Type` varchar(65) NOT NULL,
  `Check_Book` varchar(45) DEFAULT NULL,
  KEY `Account_ID_idx` (`Account_Id`),
  CONSTRAINT `Account_ID` FOREIGN KEY (`Account_Id`) REFERENCES `account` (`Account_Id`) ON I
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```



Indexes in Table

Key	Type	Uni...	Columns
Account_ID_idx	BTREE	NO	Account_Id

Index Details

Key Name:
Index Type: Packed:
Allows NULL: Unique:
Cardinality:
Comment:
User Comment:

Columns in table

Column	Type	Nullable	Indexes
Account_Id	int(11)	NO	Account_ID_idx
Account_Type	varchar(65)	NO	
Check_Book	varchar(45)	YES	

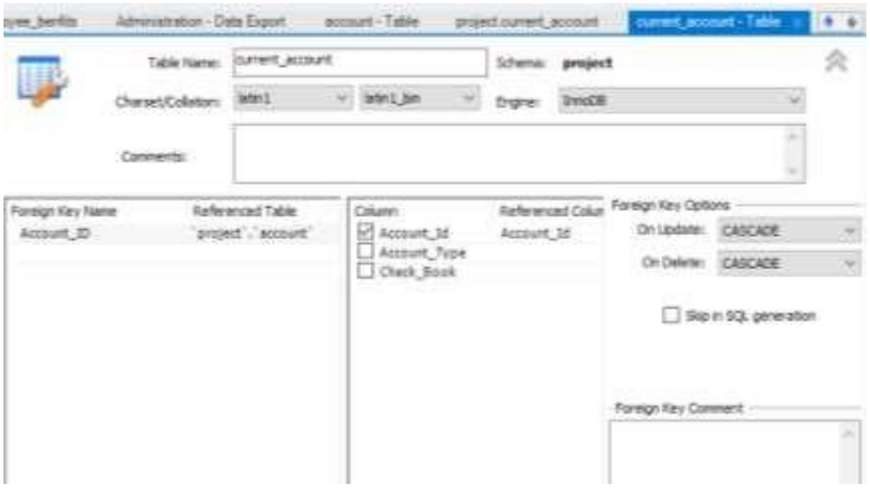
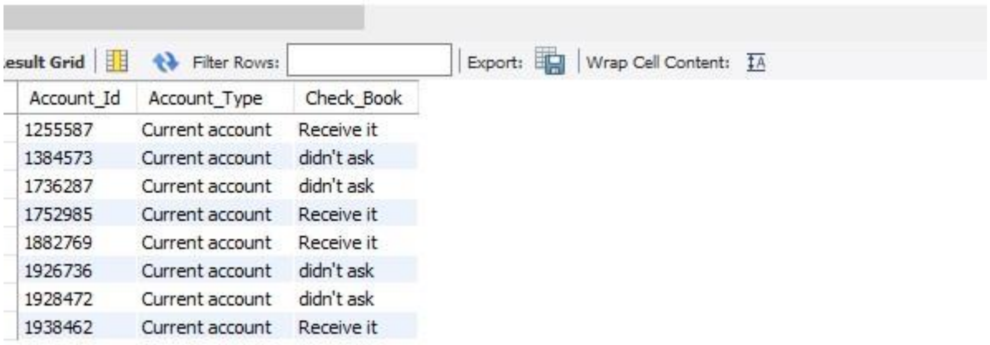


Table Name: current_account Schema: **project**
Charset/Collation: latin1 latin1_bin Engine: InnoDB
Comments:

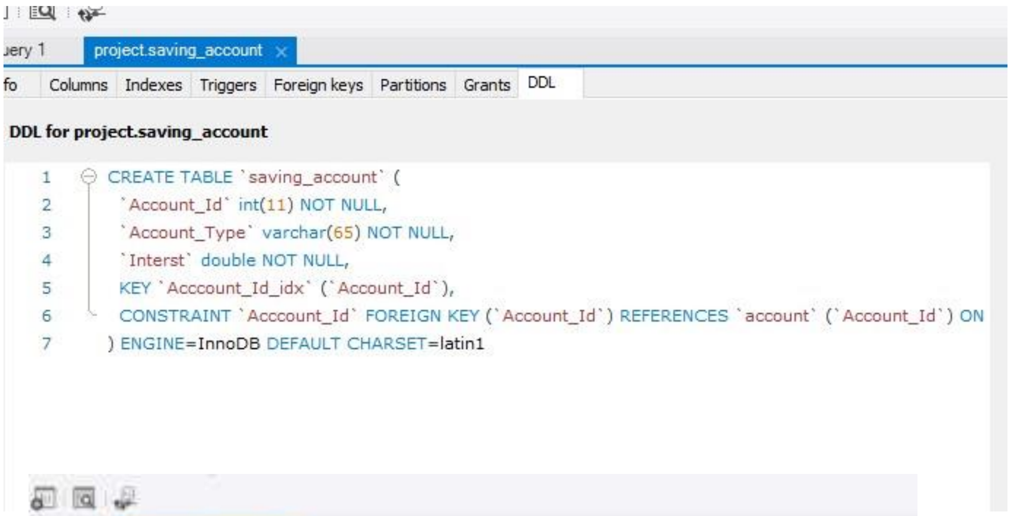
Foreign Key Name	Referenced Table
Account_ID	`project`.`account`

Column	Referenced Colur
☑ Account_Id	Account_Id
☐ Account_Type	
☐ Check_Book	

Foreign Key Options
On Update: CASCADE
On Delete: CASCADE
☐ Skip in SQL generation
Foreign Key Comment

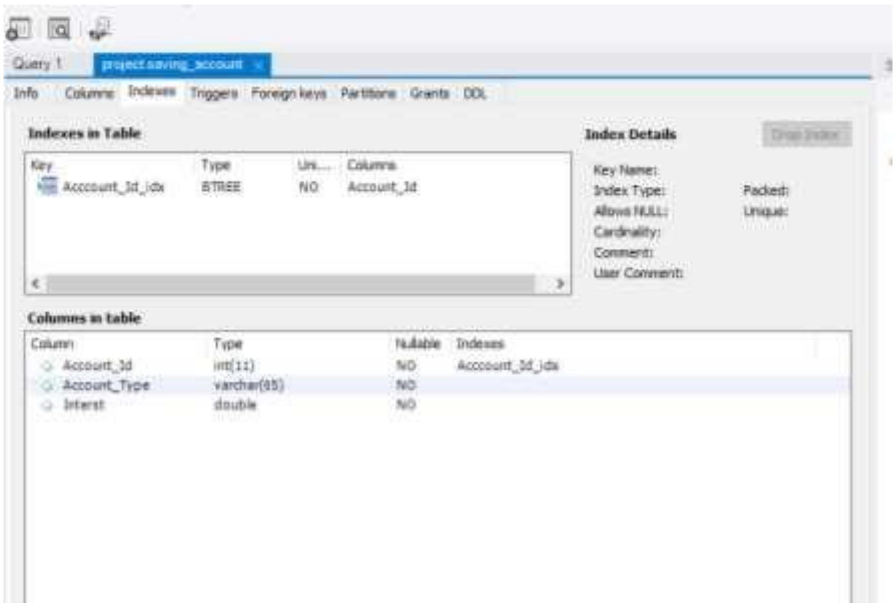


Account_Id	Account_Type	Check_Book
1255587	Current account	Receive it
1384573	Current account	didn't ask
1736287	Current account	didn't ask
1752985	Current account	Receive it
1882769	Current account	Receive it
1926736	Current account	didn't ask
1928472	Current account	didn't ask
1938462	Current account	Receive it



DDL for project.saving_account

```
CREATE TABLE `saving_account` (
  `Account_Id` int(11) NOT NULL,
  `Account_Type` varchar(65) NOT NULL,
  `Interst` double NOT NULL,
  KEY `Acccount_Id_idx` (`Account_Id`),
  CONSTRAINT `Acccount_Id` FOREIGN KEY (`Account_Id`) REFERENCES `account` (`Account_Id`) ON
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```



Indexes in Table

Key	Type	Uni...	Columns
Acccount_Id_idx	BTREE	NO	Account_Id

Index Details

Key Name:
Index Type: Packed:
Allows NULL: Unique:
Cardinality:
Comment:
User Comment:

Columns in table

Column	Type	Nullable	Indexes
Account_Id	int(11)	NO	Acccount_Id_idx
Account_Type	varchar(65)	NO	
Interst	double	NO	

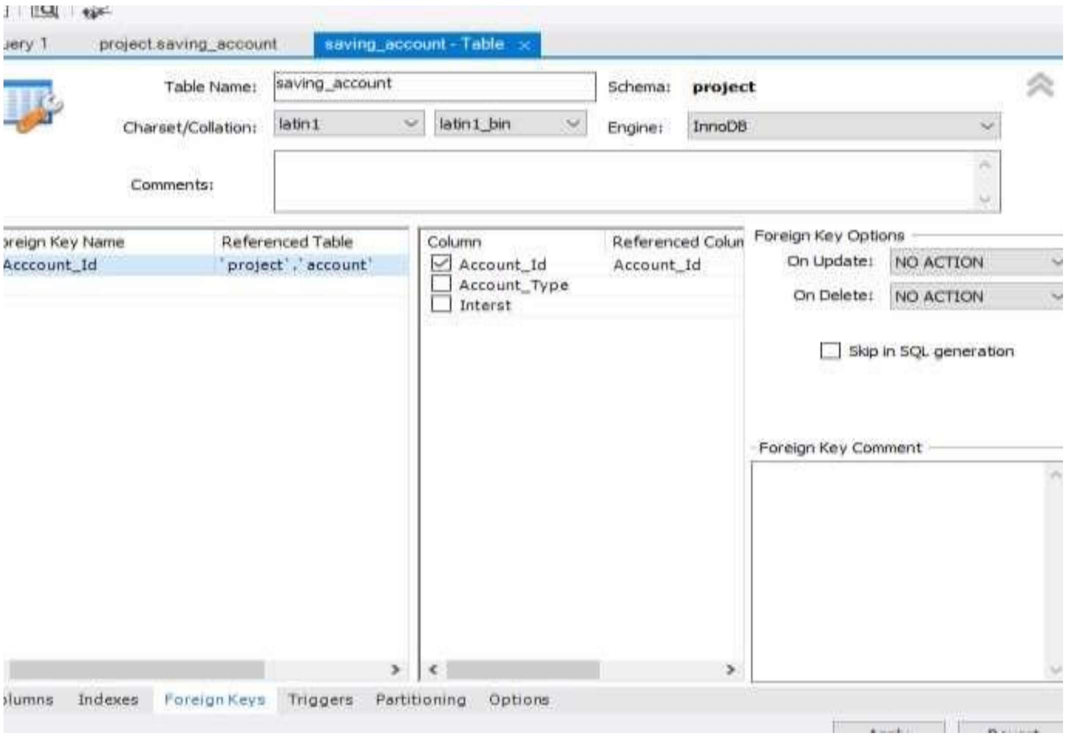


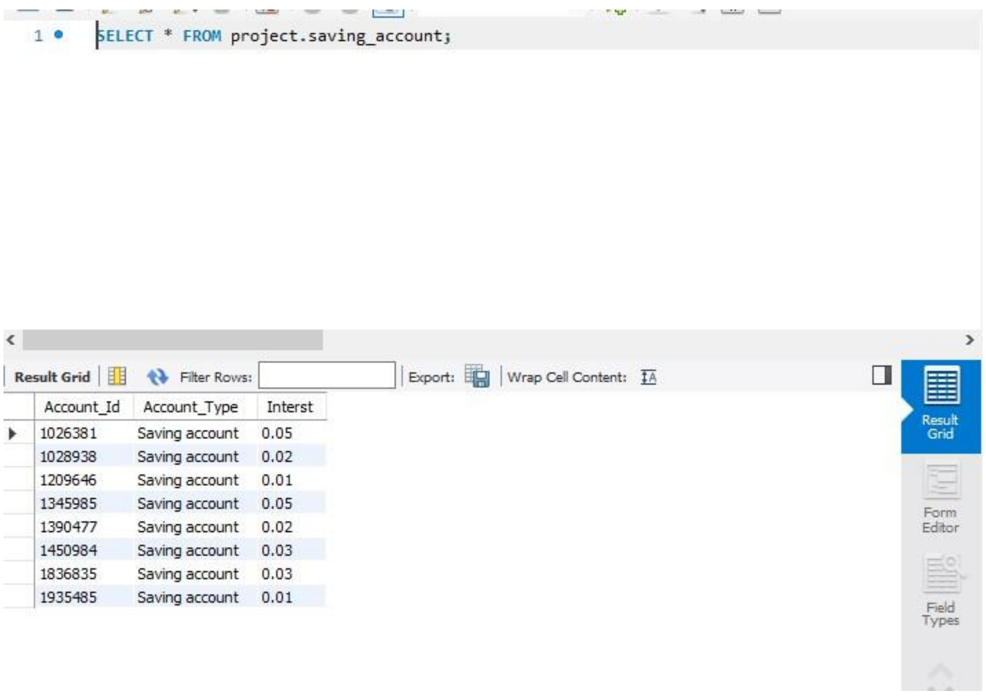
Table Name: saving_account
Schema: project
Charset/Collation: latin1 latin1_bin
Engine: InnoDB
Comments:

Foreign Key Name	Referenced Table
Acccount_Id	'project'.'account'

Column	Referenced Column
✓ Account_Id	Account_Id
Account_Type	
Interst	

Foreign Key Options
On Update: NO ACTION
On Delete: NO ACTION
Skip in SQL generation

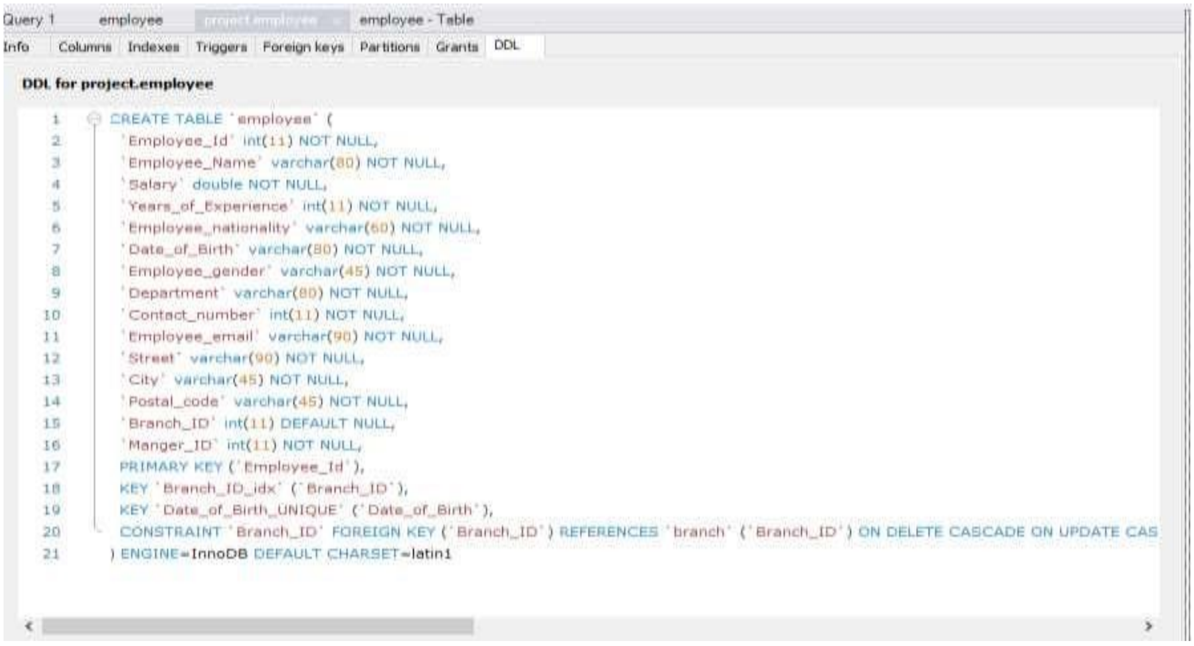
Foreign Key Comment



```
SELECT * FROM project.saving_account;
```

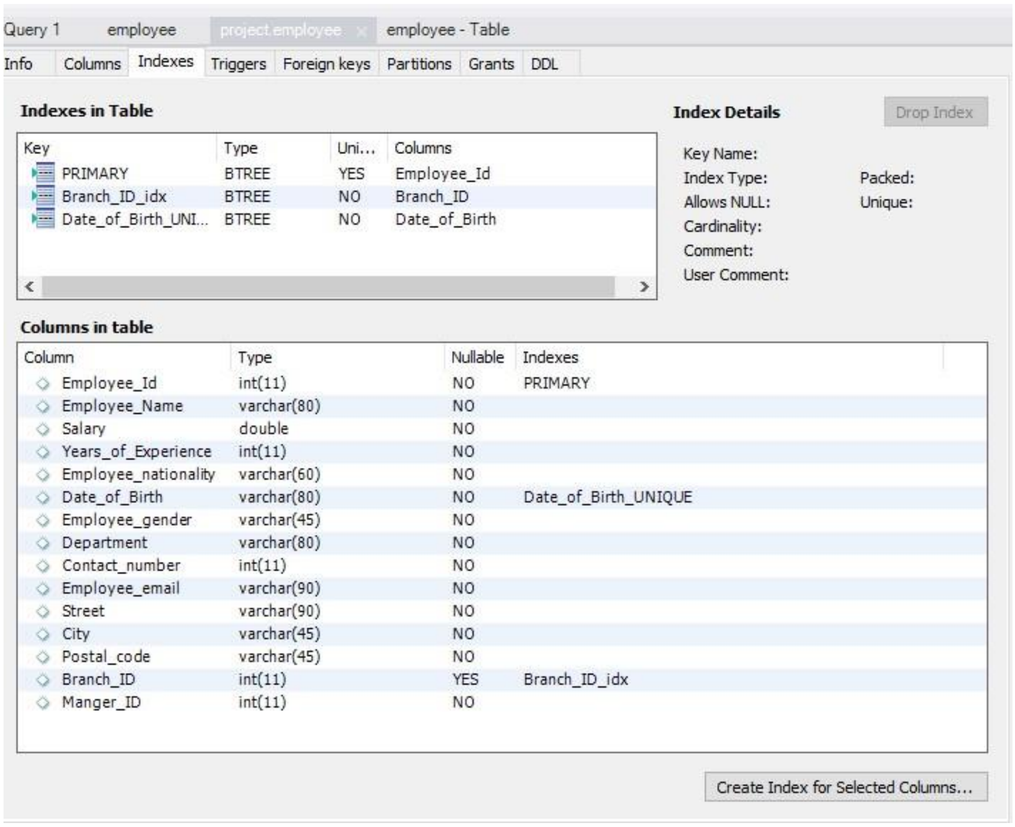
	Account_Id	Account_Type	Interst
▶	1026381	Saving account	0.05
	1028938	Saving account	0.02
	1209646	Saving account	0.01
	1345985	Saving account	0.05
	1390477	Saving account	0.02
	1450984	Saving account	0.03
	1836835	Saving account	0.03
	1935485	Saving account	0.01

Employee Table



DDL for project.employee

```
CREATE TABLE `employee` (
  `Employee_Id` int(11) NOT NULL,
  `Employee_Name` varchar(80) NOT NULL,
  `Salary` double NOT NULL,
  `Years_of_Experience` int(11) NOT NULL,
  `Employee_nationality` varchar(60) NOT NULL,
  `Date_of_Birth` varchar(80) NOT NULL,
  `Employee_gender` varchar(45) NOT NULL,
  `Department` varchar(80) NOT NULL,
  `Contact_number` int(11) NOT NULL,
  `Employee_email` varchar(90) NOT NULL,
  `Street` varchar(90) NOT NULL,
  `City` varchar(45) NOT NULL,
  `Postal_code` varchar(45) NOT NULL,
  `Branch_ID` int(11) DEFAULT NULL,
  `Manger_ID` int(11) NOT NULL,
  PRIMARY KEY (`Employee_Id`),
  KEY `Branch_ID_idx` (`Branch_ID`),
  KEY `Date_of_Birth_UNIQUE` (`Date_of_Birth`),
  CONSTRAINT `Branch_ID` FOREIGN KEY (`Branch_ID`) REFERENCES `branch` (`Branch_ID`) ON DELETE CASCADE ON UPDATE CAS
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```



Indexes in Table

Key	Type	Uni...	Columns
PRIMARY	BTREE	YES	Employee_Id
Branch_ID_idx	BTREE	NO	Branch_ID
Date_of_Birth_UNI...	BTREE	NO	Date_of_Birth

Index Details

Key Name:	
Index Type:	Packed:
Allows NULL:	Unique:
Cardinality:	
Comment:	
User Comment:	

Columns in table

Column	Type	Nullable	Indexes
Employee_Id	int(11)	NO	PRIMARY
Employee_Name	varchar(80)	NO	
Salary	double	NO	
Years_of_Experience	int(11)	NO	
Employee_nationality	varchar(60)	NO	
Date_of_Birth	varchar(80)	NO	Date_of_Birth_UNIQUE
Employee_gender	varchar(45)	NO	
Department	varchar(80)	NO	
Contact_number	int(11)	NO	
Employee_email	varchar(90)	NO	
Street	varchar(90)	NO	
City	varchar(45)	NO	
Postal_code	varchar(45)	NO	
Branch_ID	int(11)	YES	Branch_ID_idx
Manger_ID	int(11)	NO	

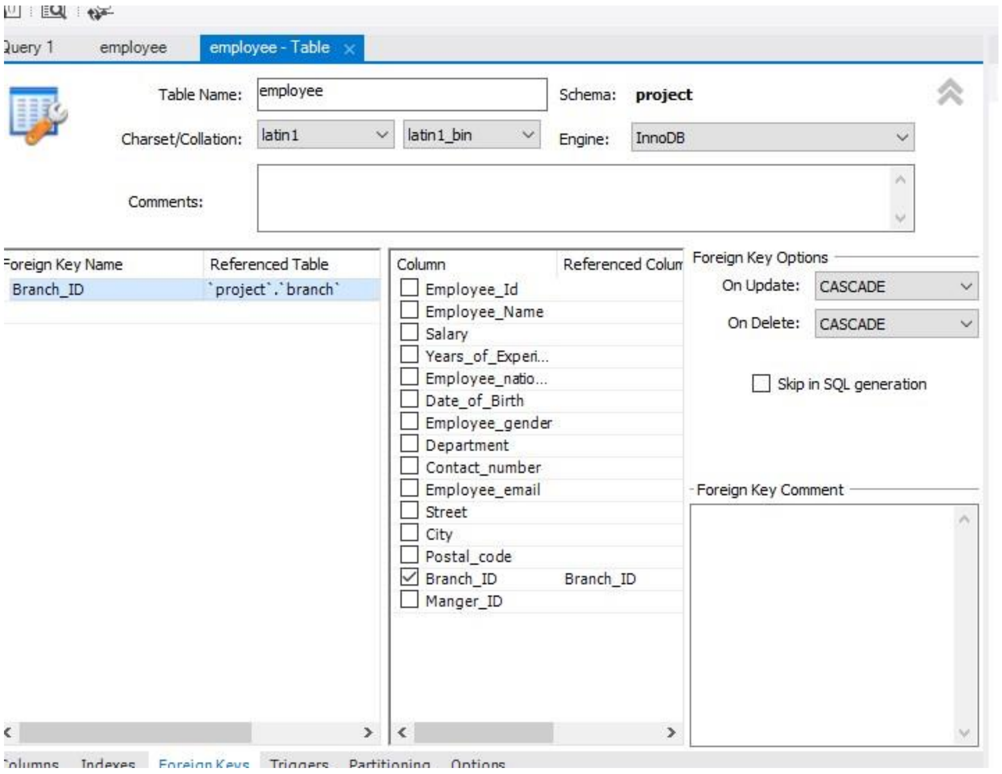
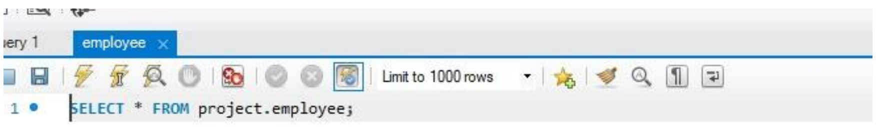


Table Name: employee
Schema: project
Charset/Collation: latin1 latin1_bin
Engine: InnoDB
Comments:

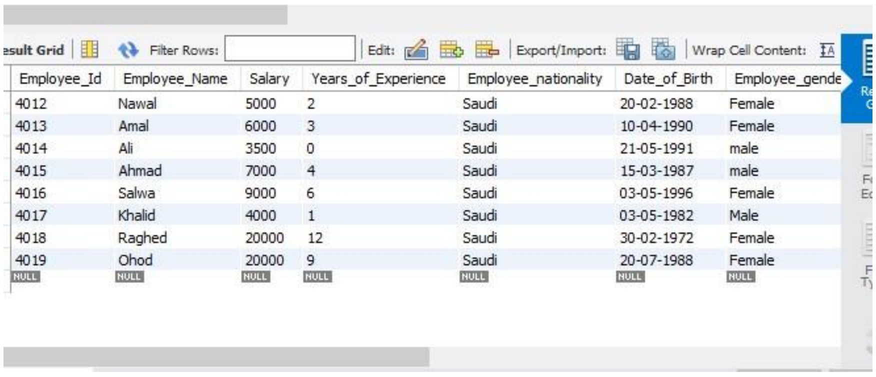
Foreign Key Name	Referenced Table
Branch_ID	`project`.`branch`

Column	Referenced Column
☐ Employee_Id	
☐ Employee_Name	
☐ Salary	
☐ Years_of_Experi...	
☐ Employee_natio...	
☐ Date_of_Birth	
☐ Employee_gender	
☐ Department	
☐ Contact_number	
☐ Employee_email	
☐ Street	
☐ City	
☐ Postal_code	
✓ Branch_ID	Branch_ID
☐ Manger_ID	

Foreign Key Options
On Update: CASCADE
On Delete: CASCADE
☐ Skip in SQL generation
Foreign Key Comment

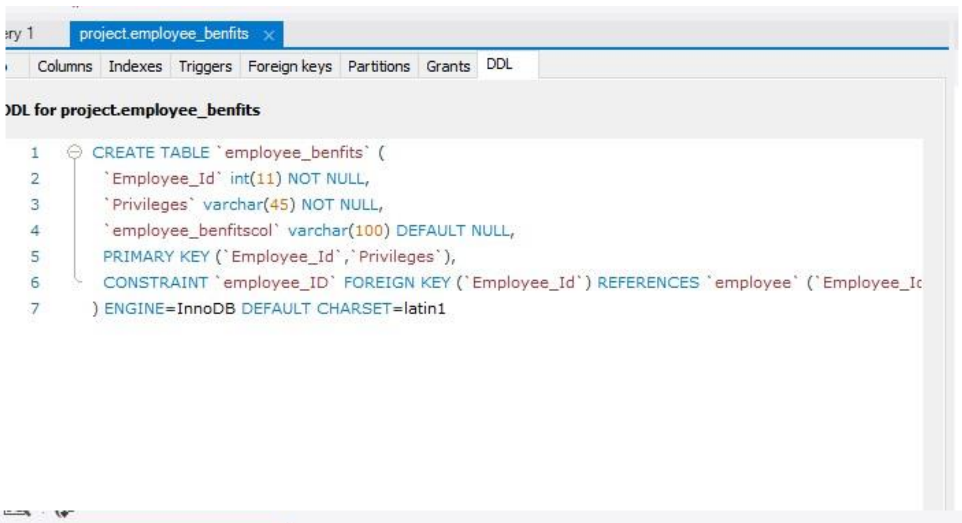


```
SELECT * FROM project.employee;
```



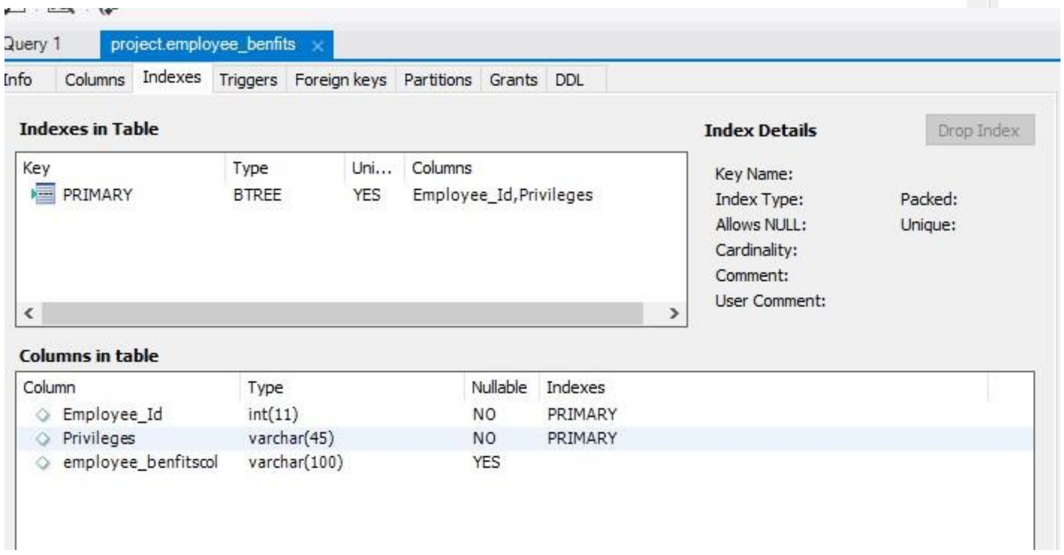
Employee_Id	Employee_Name	Salary	Years_of_Experience	Employee_nationality	Date_of_Birth	Employee_gende
4012	Nawal	5000	2	Saudi	20-02-1988	Female
4013	Amal	6000	3	Saudi	10-04-1990	Female
4014	Ali	3500	0	Saudi	21-05-1991	male
4015	Ahmad	7000	4	Saudi	15-03-1987	male
4016	Salwa	9000	6	Saudi	03-05-1996	Female
4017	Khalid	4000	1	Saudi	03-05-1982	Male
4018	Raghed	20000	12	Saudi	30-02-1972	Female
4019	Ohod	20000	9	Saudi	20-07-1988	Female
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Employee benefits Table



DDL for project.employee_benfits

```
CREATE TABLE `employee_benfits` (
  `Employee_Id` int(11) NOT NULL,
  `Privileges` varchar(45) NOT NULL,
  `employee_benfitscol` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Employee_Id`,`Privileges`),
  CONSTRAINT `employee_ID` FOREIGN KEY (`Employee_Id`) REFERENCES `employee` (`Employee_Ic
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```



Indexes in Table

Key	Type	Uni...	Columns
PRIMARY	BTREE	YES	Employee_Id,Privileges

Index Details

Key Name:
Index Type: Packed:
Allows NULL: Unique:
Cardinality:
Comment:
User Comment:

Columns in table

Column	Type	Nullable	Indexes
Employee_Id	int(11)	NO	PRIMARY
Privileges	varchar(45)	NO	PRIMARY
employee_benfitscol	varchar(100)	YES	

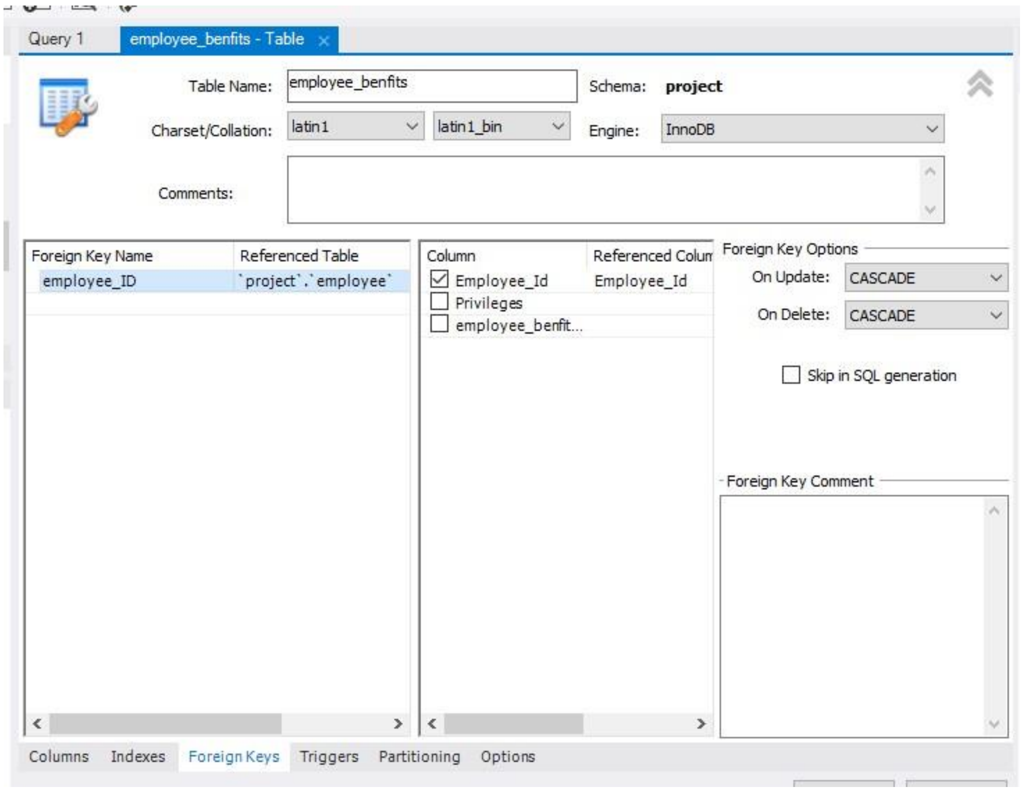
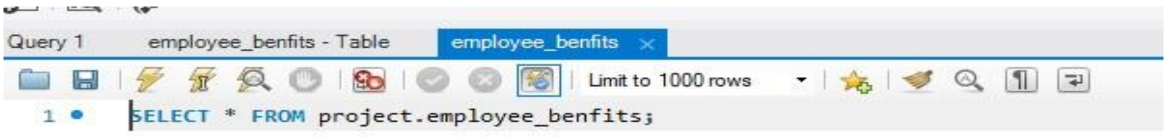


Table Name: employee_benfits
Schema: project
Charset/Collation: latin1 latin1_bin
Engine: InnoDB
Comments:

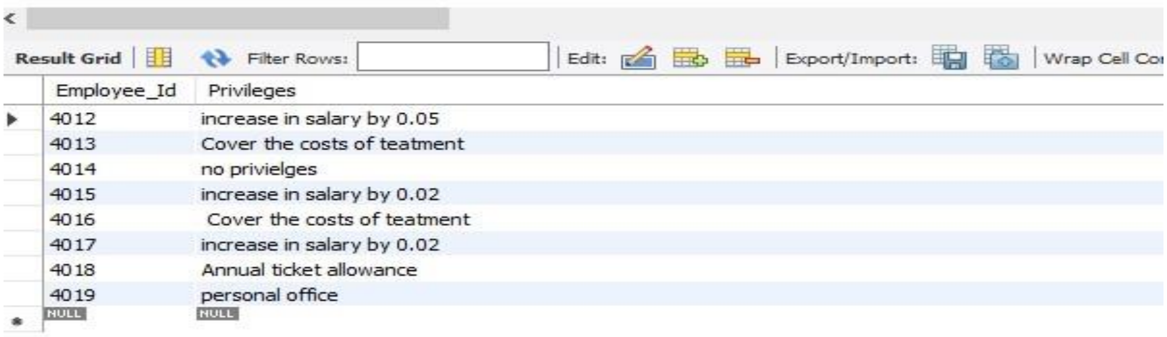
Foreign Key Name	Referenced Table
employee_ID	`project`.`employee`

Column	Referenced Column
✓ Employee_Id	Employee_Id
Privileges	
employee_benfit...	

Foreign Key Options
On Update: CASCADE
On Delete: CASCADE
Skip in SQL generation
Foreign Key Comment



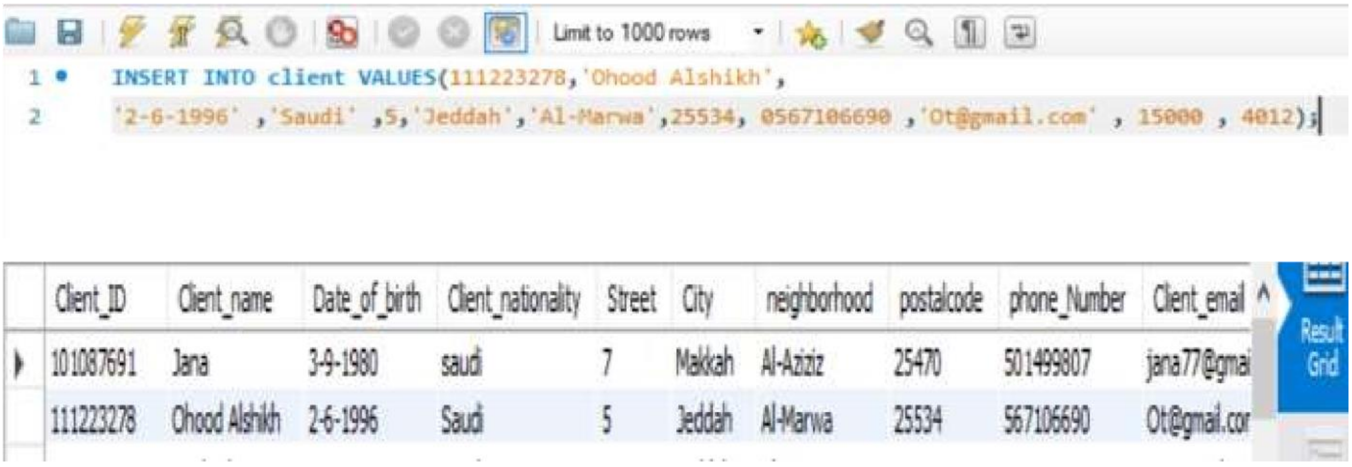
```
SELECT * FROM project.employee_benfits;
```



Employee_Id	Privileges
4012	increase in salary by 0.05
4013	Cover the costs of teatment
4014	no privielges
4015	increase in salary by 0.02
4016	Cover the costs of teatment
4017	increase in salary by 0.02
4018	Annual ticket allowance
4019	personal office
NULL	NULL

3.2 Queries

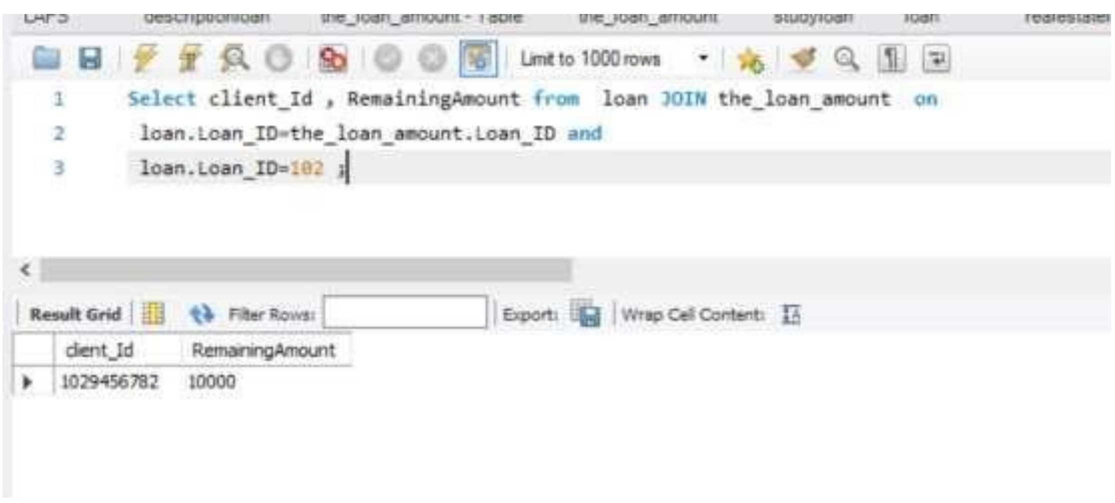
Insert new client



```
INSERT INTO client VALUES(111223278,'Ohood Alshikh',
'2-6-1996' ,'Saudi' ,5,'Jeddah','Al-Marwa',25534, 0567106690 ,'Ot@gmail.com' , 15000 , 4012);
```

Client_ID	Client_name	Date_of_birth	Client_nationality	Street	City	neighborhood	postalcode	phone_Number	Client_email
101087691	Jana	3-9-1980	saudi	7	Makkah	Al-Aziziz	25470	501499807	jana77@gmai
111223278	Ohood Alshikh	2-6-1996	Saudi	5	Jeddah	Al-Marwa	25534	567106690	Ot@gmail.cor

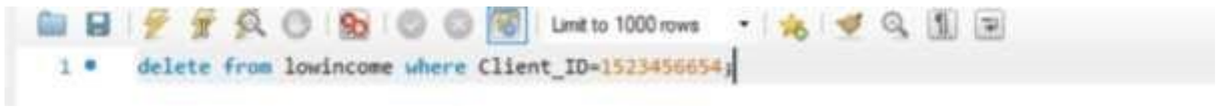
Display the ID of Client and the Remaining Amount of Loan that its ID =102 .



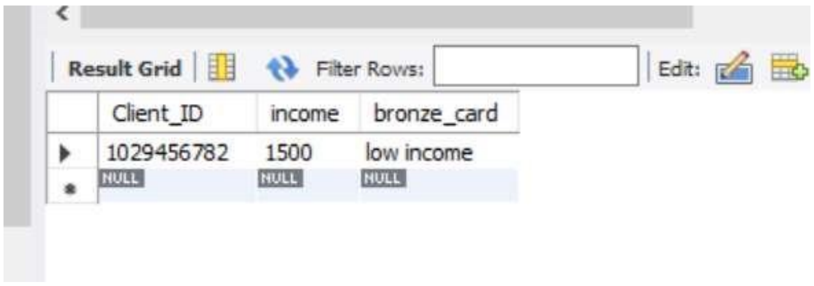
```
Select client_Id , RemainingAmount from  loan JOIN the_loan_amount  on
 loan.Loan_ID=the_loan_amount.Loan_ID and
 loan.Loan_ID=102 ;
```

	client_Id	RemainingAmount
▶	1029456782	10000

Delete



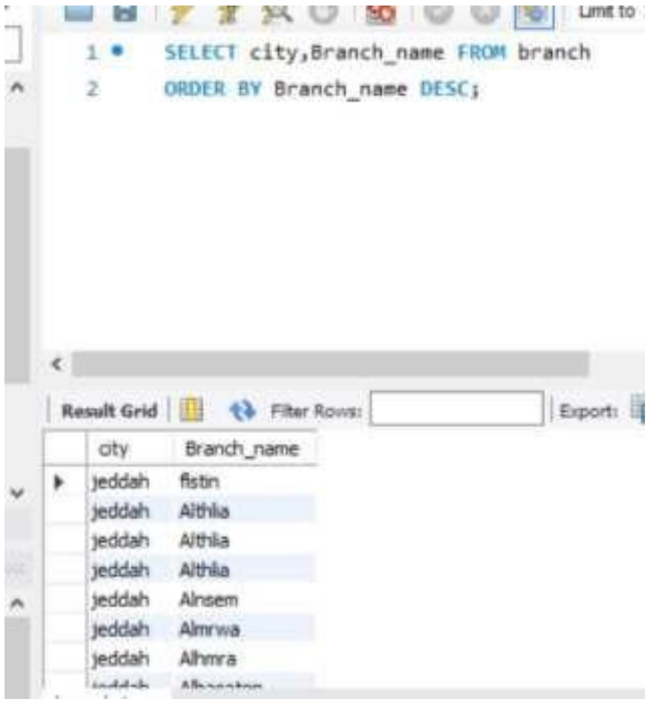
```
delete from lowincome where Client_ID=1523456654;
```



	Client_ID	income	bronze_card
▶	1029456782	1500	low income
*	NULL	NULL	NULL

Delete from table low income the client with 1523456654 id

Order by

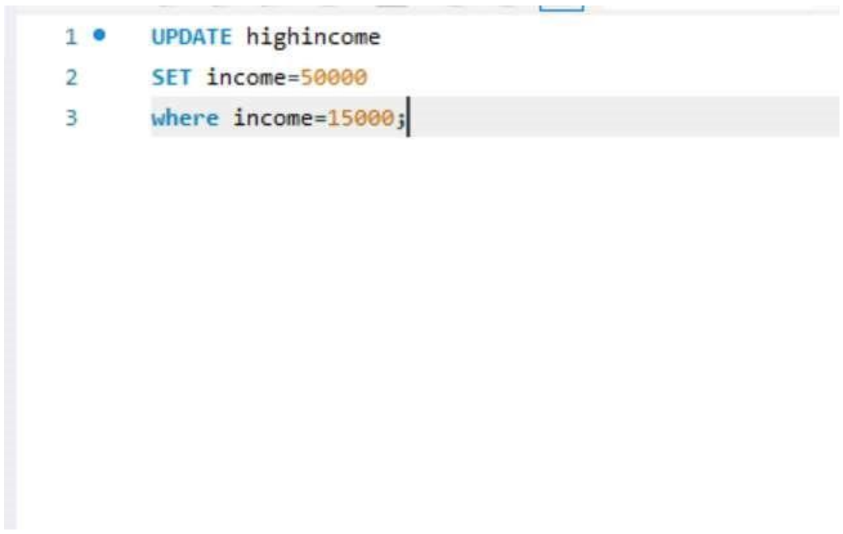


```
SELECT city,Branch_name FROM branch
ORDER BY Branch_name DESC;
```

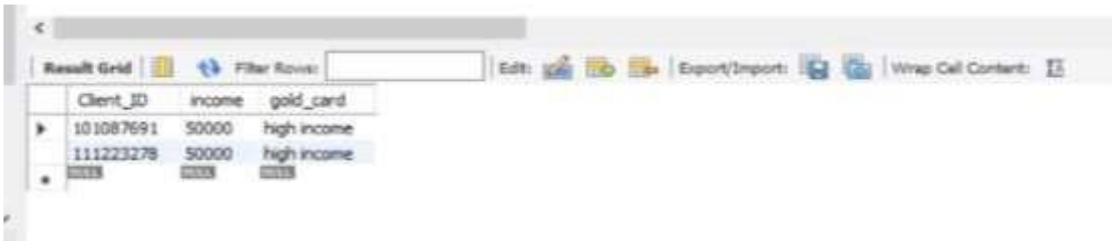
city	Branch_name
jeddah	fistin
jeddah	Althlia
jeddah	Althlia
jeddah	Althlia
jeddah	Alnsem
jeddah	Almrwa
jeddah	Alhmra

Order the branch name by DESC from A-Z

Update



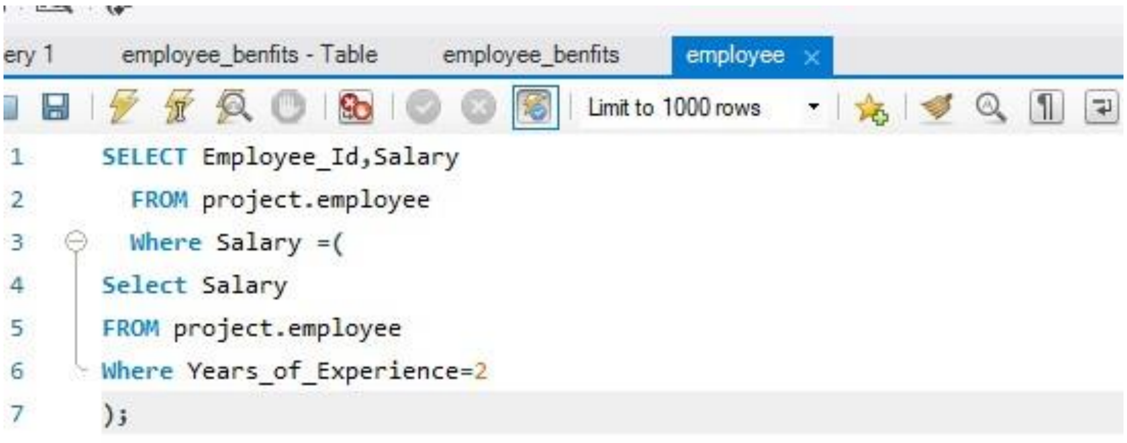
```
UPDATE highincome
SET income=50000
where income=15000;
```



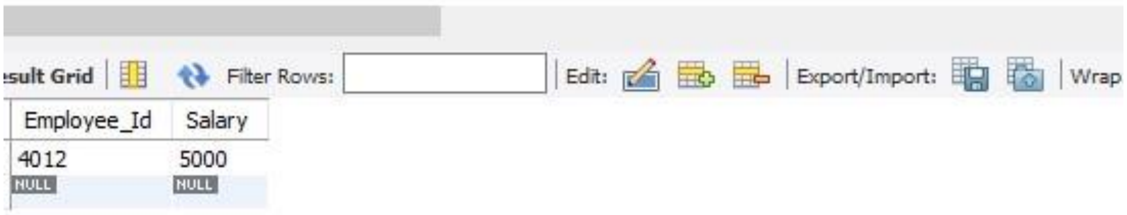
Client_ID	income	gold_card
101087691	50000	high income
111223278	50000	high income
NULL	NULL	NULL

Change income from 15000 to 50000

Display employee id and salary of employee whose years of experience =2



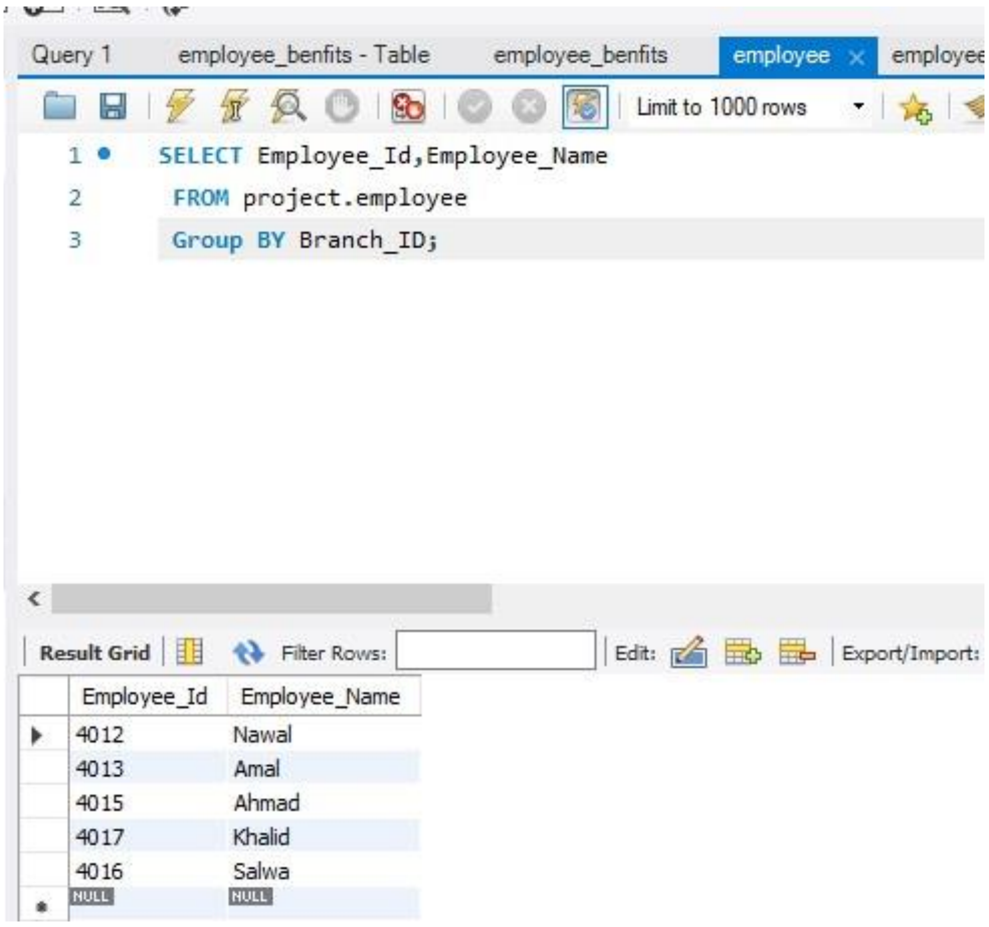
```
SELECT Employee_Id,Salary
  FROM project.employee
  Where Salary =(
Select Salary
FROM project.employee
Where Years_of_Experience=2
);
```



Employee_Id	Salary
4012	5000
NULL	NULL

Display employee's id and name of

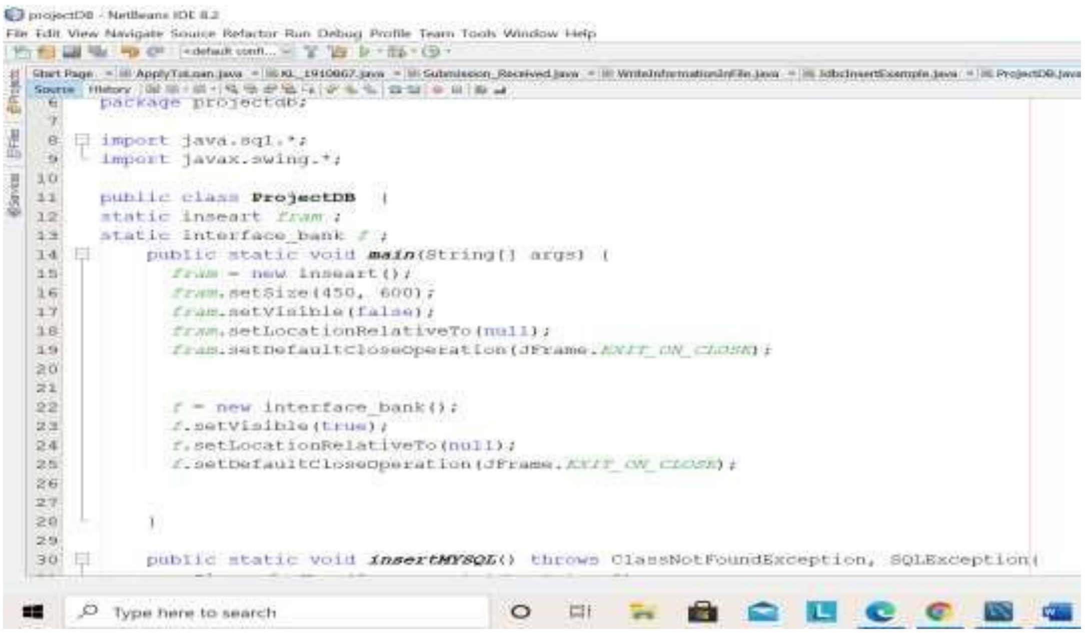
each branch



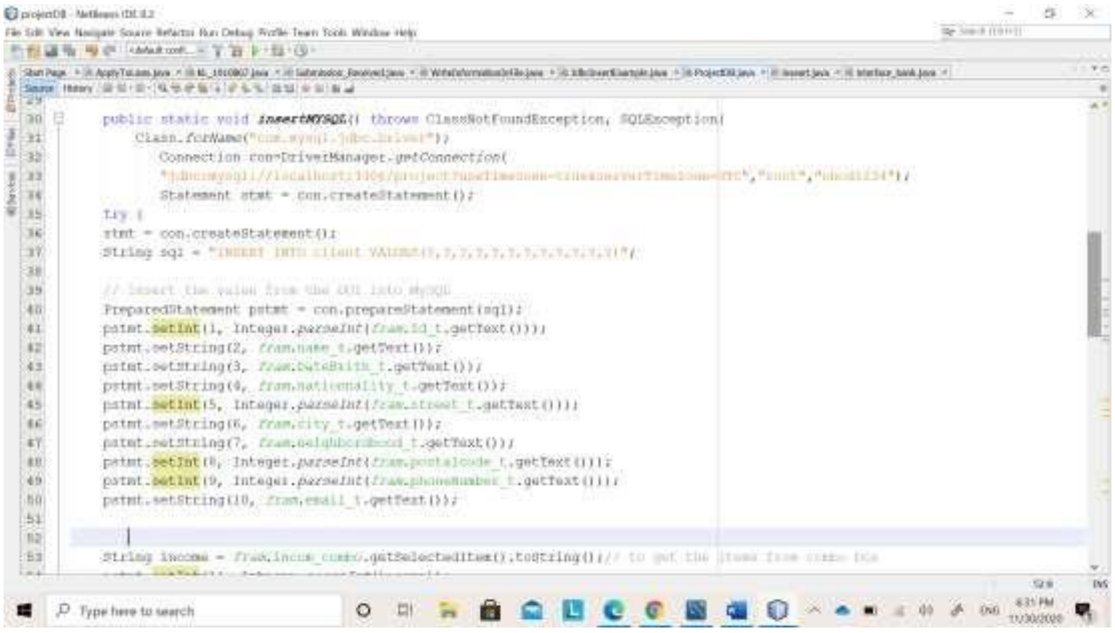
```
SELECT Employee_Id,Employee_Name
 FROM project.employee
 Group BY Branch_ID;
```

Employee_Id	Employee_Name
4012	Nawal
4013	Amal
4015	Ahmad
4017	Khalid
4016	Salwa
NULL	NULL

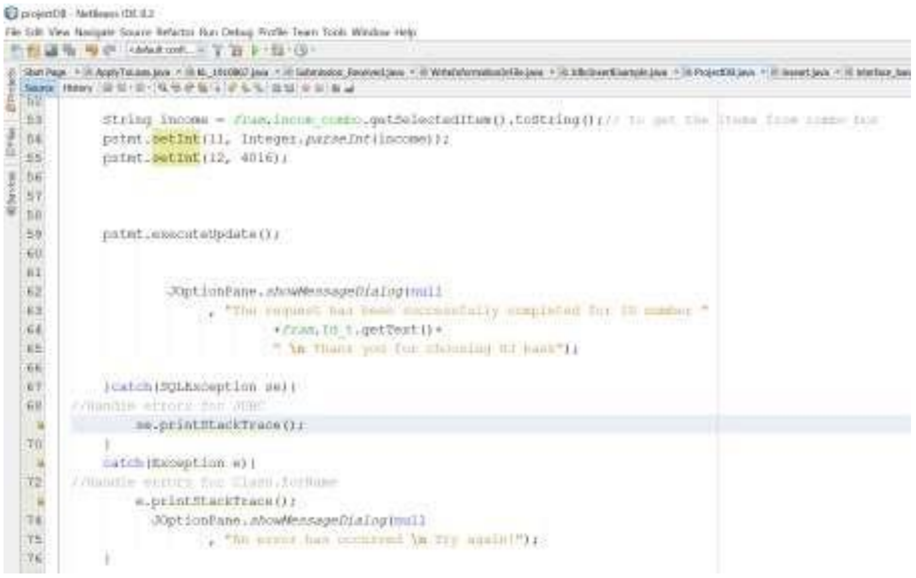
3.3 JBC code



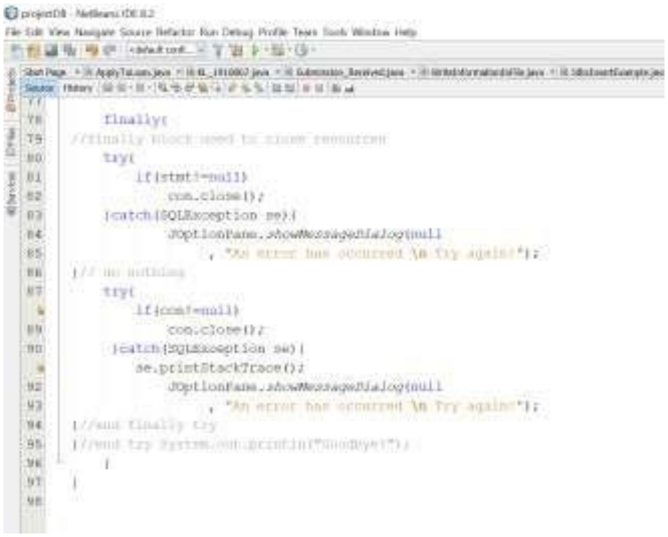
```java
package projectdb;

import java.sql.*;
import javax.swing.*;

public class ProjectDB  {
static inseart fram ;
static interface_bank f ;
    public static void main(String[] args) {
      fram = new inseart();
      fram.setSize(450, 600);
      fram.setVisible(false);
      fram.setLocationRelativeTo(null);
      fram.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


      f = new interface_bank();
      f.setVisible(true);
      f.setLocationRelativeTo(null);
      f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


    }

    public static void insertMYSQL() throws ClassNotFoundException, SQLException{
```



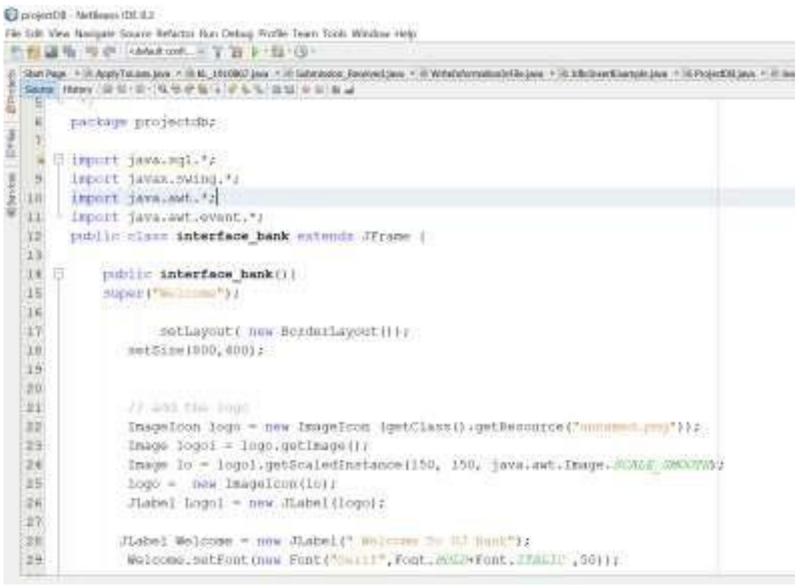
```java
    public static void insertMYSQL() throws ClassNotFoundException, SQLException{
        Class.forName("com.mysql.jdbc.Driver");
            Connection con=DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/project?useTimezone=true&serverTimezone=UTC","root","abcd1234");
            Statement stmt = con.createStatement();
    try {
    stmt = con.createStatement();
    String sql = "INSERT INTO client VALUES(?,?,?,?,?,?,?,?,?,?,?,?)";

    // Insert the value from the GUI into MySQL
    PreparedStatement pstmt = con.prepareStatement(sql);
    pstmt.setInt(1, Integer.parseInt(fram.Id_t.getText()));
    pstmt.setString(2, fram.name_t.getText());
    pstmt.setString(3, fram.DateBirth_t.getText());
    pstmt.setString(4, fram.nationality_t.getText());
    pstmt.setInt(5, Integer.parseInt(fram.street_t.getText()));
    pstmt.setString(6, fram.city_t.getText());
    pstmt.setString(7, fram.neighborhood_t.getText());
    pstmt.setInt(8, Integer.parseInt(fram.postalcode_t.getText()));
    pstmt.setInt(9, Integer.parseInt(fram.phoneNumber_t.getText()));
    pstmt.setString(10, fram.email_t.getText());


    String income = fram.incum_combo.getSelectedItem().toString();// to get the items from combo box
```



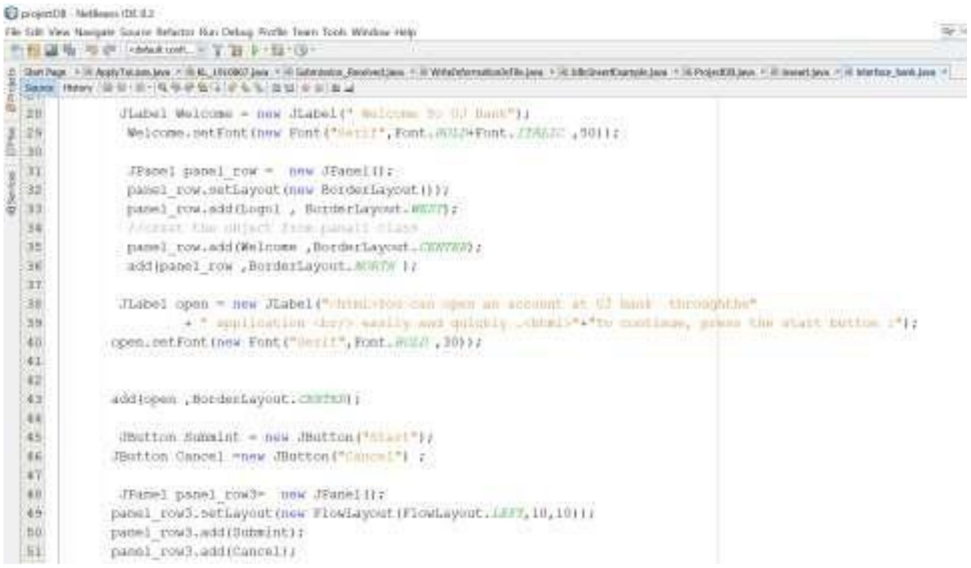
```
    String income = fram.incom_combo.getSelectedItem().toString();// to get the items from combo box
    pstmt.setInt(11, Integer.parseInt(income));
    pstmt.setInt(12, 4016);



    pstmt.executeUpdate();



            JOptionPane.showMessageDialog(null
                , "The request has been successfully completed for ID number "
                        +fram.Id_t.getText()+
                        " \n Thank you for choosing HJ bank");

    }catch(SQLException se){
//Handle errors for JDBC
        se.printStackTrace();
    }
    catch(Exception e){
//Handle errors for Class.forName
        e.printStackTrace();
          JOptionPane.showMessageDialog(null
                , "An error has occurred \n Try again!");
    }
```



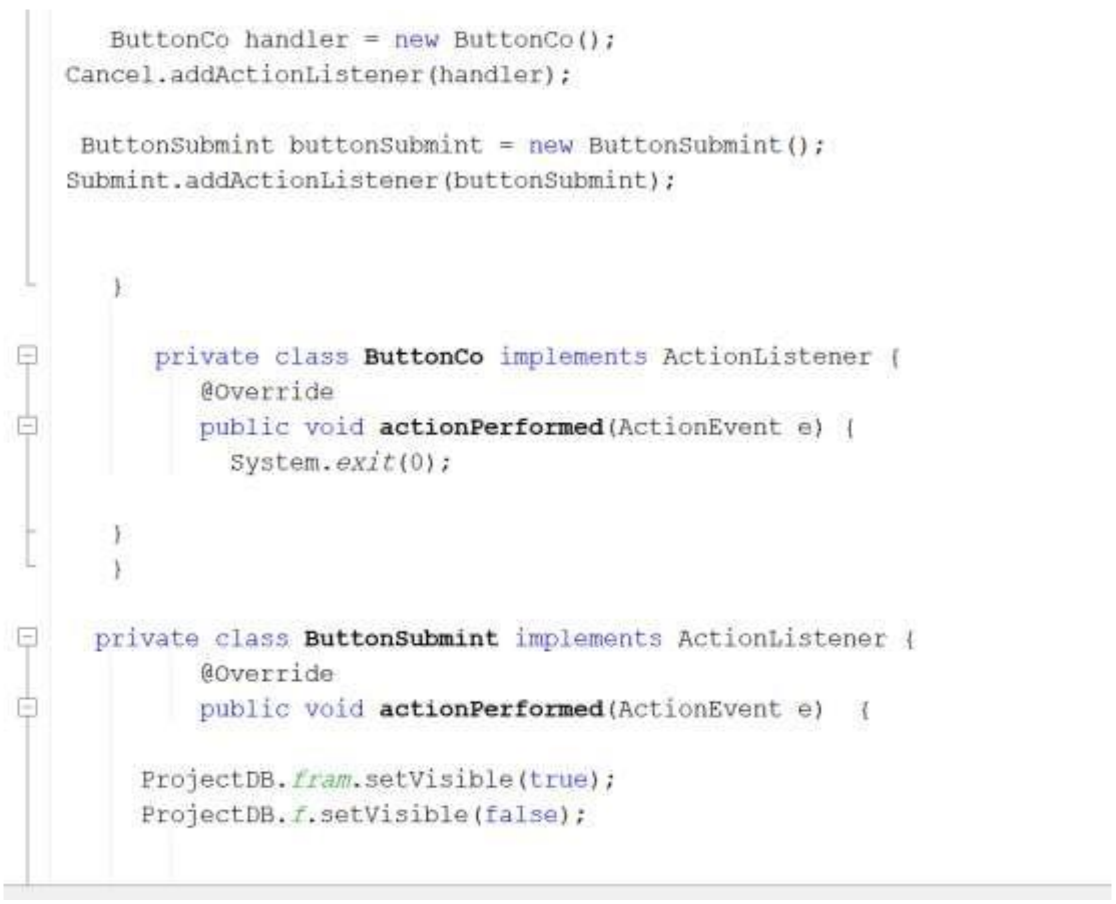
```
    finally{
//finally block used to close resources
    try{
        if(stmt!=null)
            con.close();
    }catch(SQLException se){
            JOptionPane.showMessageDialog(null
                , "An error has occurred \n Try again!");
}// do nothing
    try{
        if(con!=null)
            con.close();
     }catch(SQLException se){
        se.printStackTrace();
            JOptionPane.showMessageDialog(null
                , "An error has occurred \n Try again!");
}//end finally try
}//end try System.out.println("Goodbye!");
    }
}
```



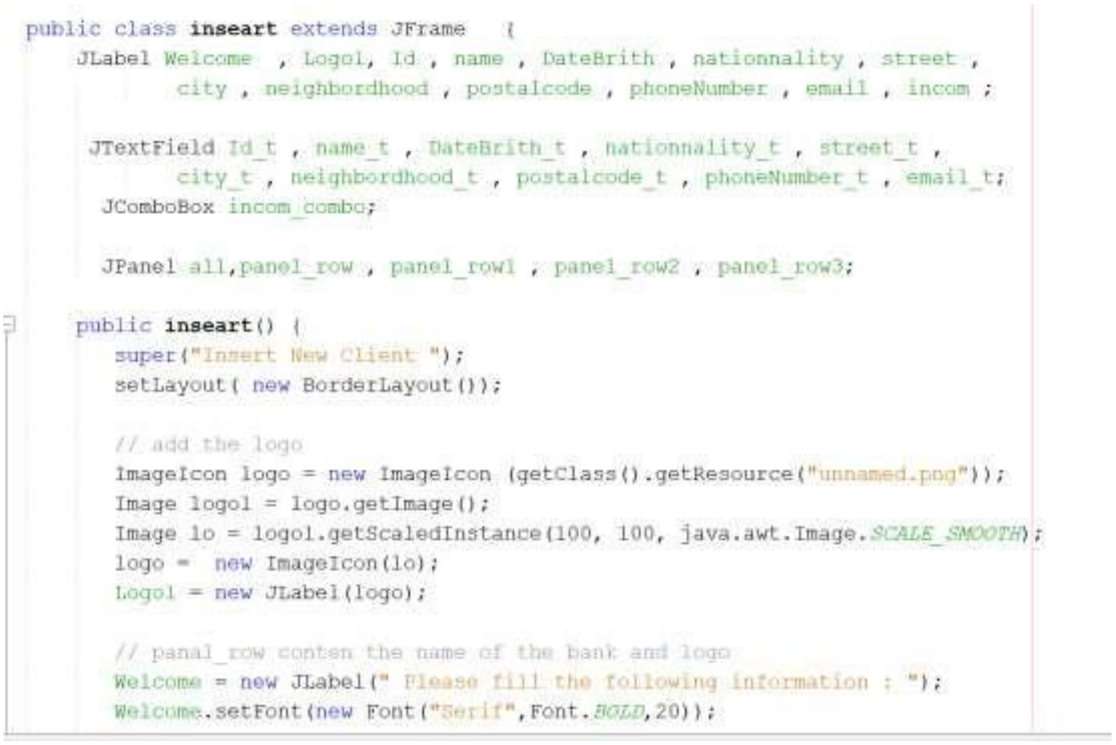
```
package projectdb;

import java.sql.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class interface_bank extends JFrame {

    public interface_bank(){
    super("Welcome");

          setLayout( new BorderLayout());
      setSize(800,400);


      // add the logo
      ImageIcon logo = new ImageIcon (getClass().getResource("unnamed.png"));
      Image logo1 = logo.getImage();
      Image lo = logo1.getScaledInstance(150, 150, java.awt.Image.SCALE_SMOOTH);
      logo =  new ImageIcon(lo);
      JLabel Logo1 = new JLabel(logo);

     JLabel Welcome = new JLabel(" Welcome To HJ bank");
      Welcome.setFont(new Font("Serif",Font.BOLD+Font.ITALIC ,50));
```


projectDB - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Start Page
ApplyToLoan.java
Submission_Received.java
WriteInformationInFile.java
ProjectDB.java
Insert.java
Interface_bank.java
Source
History

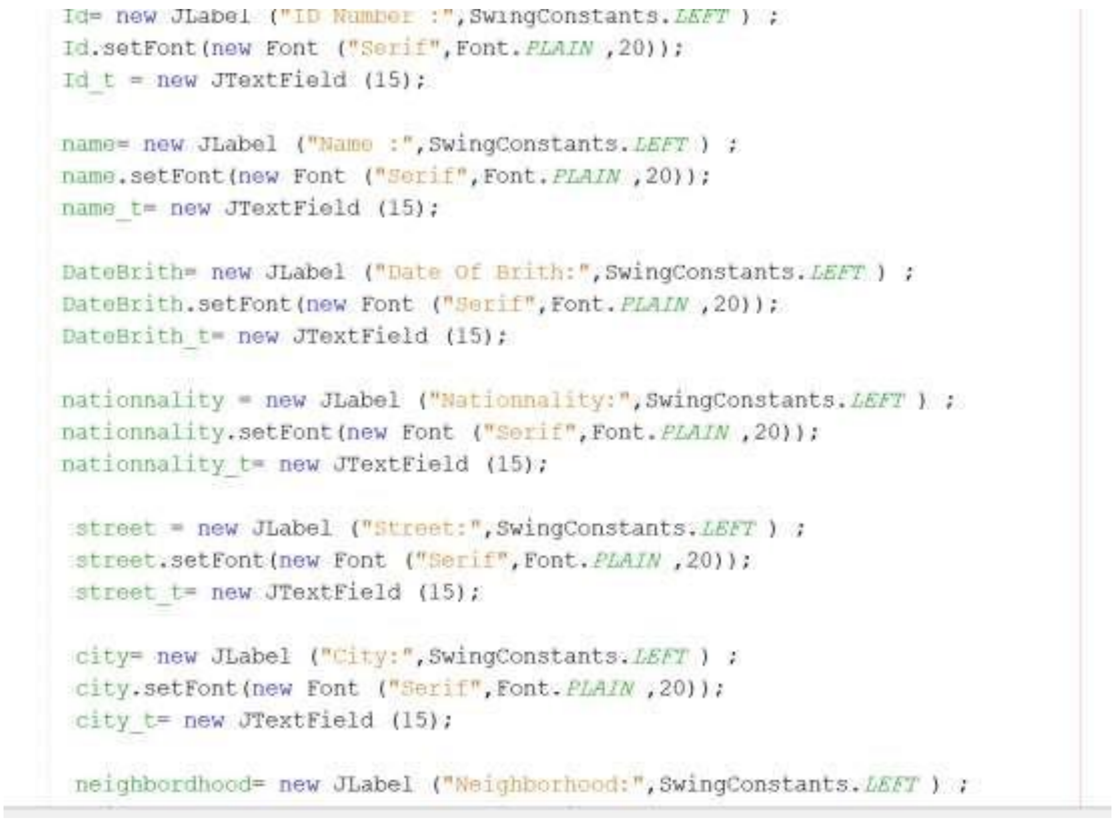
```
        JLabel Welcome = new JLabel(" Welcome To UJ Bank");
         Welcome.setFont(new Font("Serif",Font.BOLD+Font.ITALIC ,50));

         JPanel panel_row =  new JPanel();
         panel_row.setLayout(new BorderLayout());
         panel_row.add(Logo1 , BorderLayout.WEST);
         //creat the object from panel1 class
         panel_row.add(Welcome ,BorderLayout.CENTER);
         add(panel_row ,BorderLayout.NORTH );

        JLabel open = new JLabel("<html>You can open an account at UJ bank  throughthe"
                + " application <br/> easily and quickly .</html>"+"to continue, press the start button :");
       open.setFont(new Font("Serif",Font.BOLD ,30));


       add(open ,BorderLayout.CENTER);

        JButton Submint = new JButton("Start");
       JButton Cancel =new JButton("Cancel") ;

        JPanel panel_row3=  new JPanel();
       panel_row3.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));
       panel_row3.add(Submint);
       panel_row3.add(Cancel);
```



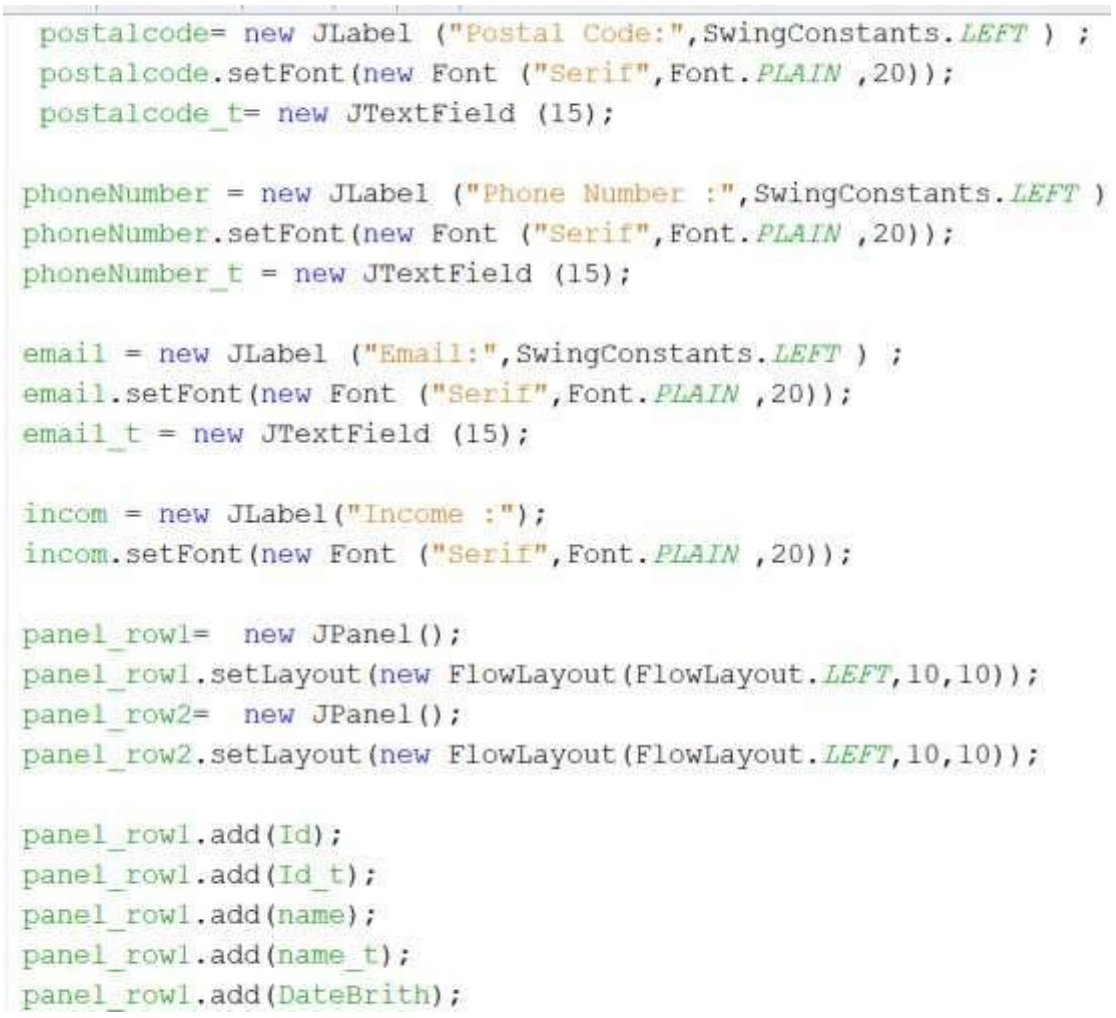
```
   ButtonCo handler = new ButtonCo();
Cancel.addActionListener(handler);

 ButtonSubmint buttonSubmint = new ButtonSubmint();
Submint.addActionListener(buttonSubmint);


      }

         private class ButtonCo implements ActionListener {
            @Override
            public void actionPerformed(ActionEvent e) {
              System.exit(0);

      }
      }

     private class ButtonSubmint implements ActionListener {
            @Override
            public void actionPerformed(ActionEvent e)   {

          ProjectDB.fram.setVisible(true);
          ProjectDB.f.setVisible(false);
```



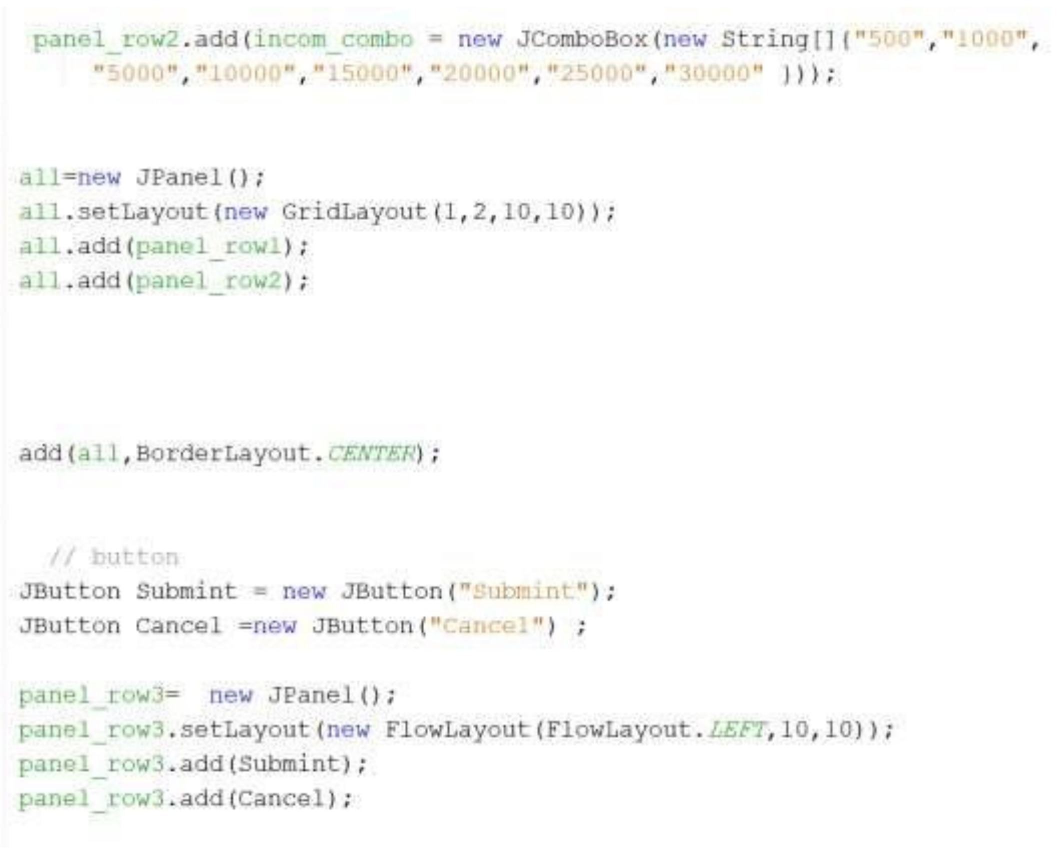
```
public class inseart extends JFrame   {
    JLabel Welcome  , Logo1, Id , name , DateBrith , nationnality , street ,
            city , neighbordhood , postalcode , phoneNumber , email , incom ;

     JTextField Id_t , name_t , DateBrith_t , nationnality_t , street_t ,
            city_t , neighbordhood_t , postalcode_t , phoneNumber_t , email_t;
      JComboBox incom_combo;

      JPanel all,panel_row , panel_row1 , panel_row2 , panel_row3;

    public inseart() {
        super("Insert New Client ");
        setLayout( new BorderLayout());

        // add the logo
        ImageIcon logo = new ImageIcon (getClass().getResource("unnamed.png"));
        Image logo1 = logo.getImage();
        Image lo = logo1.getScaledInstance(100, 100, java.awt.Image.SCALE_SMOOTH);
        logo =  new ImageIcon(lo);
        Logo1 = new JLabel(logo);

        // panal_row conten the name of the bank and logo
        Welcome = new JLabel(" Please fill the following information : ");
        Welcome.setFont(new Font("Serif",Font.BOLD,20));
```



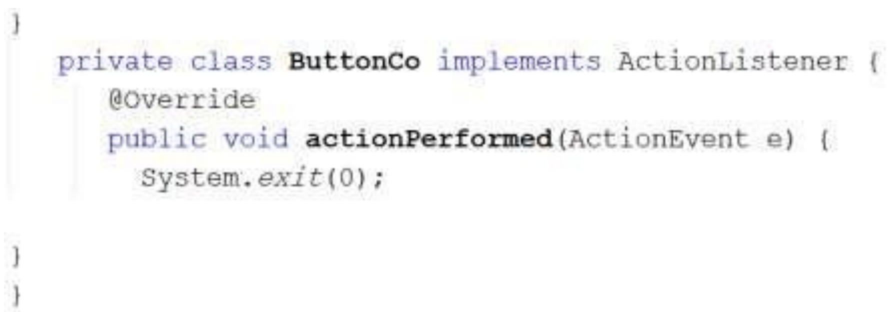
```
  Id= new JLabel  ("ID Number :",SwingConstants.LEFT ) ;
  Id.setFont(new Font  ("Serif",Font.PLAIN ,20));
  Id_t = new JTextField (15);

  name= new JLabel  ("Name :",SwingConstants.LEFT ) ;
  name.setFont(new Font  ("Serif",Font.PLAIN ,20));
  name_t= new JTextField (15);

  DateBrith= new JLabel  ("Date Of Brith:",SwingConstants.LEFT ) ;
  DateBrith.setFont(new Font  ("Serif",Font.PLAIN ,20));
  DateBrith_t= new JTextField (15);

  nationnality = new JLabel  ("Nationnality:",SwingConstants.LEFT ) ;
  nationnality.setFont(new Font  ("Serif",Font.PLAIN ,20));
  nationnality_t= new JTextField (15);

   street = new JLabel  ("Street:",SwingConstants.LEFT ) ;
   street.setFont(new Font  ("Serif",Font.PLAIN ,20));
   street_t= new JTextField (15);

   city= new JLabel  ("City:",SwingConstants.LEFT ) ;
   city.setFont(new Font  ("Serif",Font.PLAIN ,20));
   city_t= new JTextField (15);

   neighbordhood= new JLabel  ("Neighborhood:",SwingConstants.LEFT ) ;
```



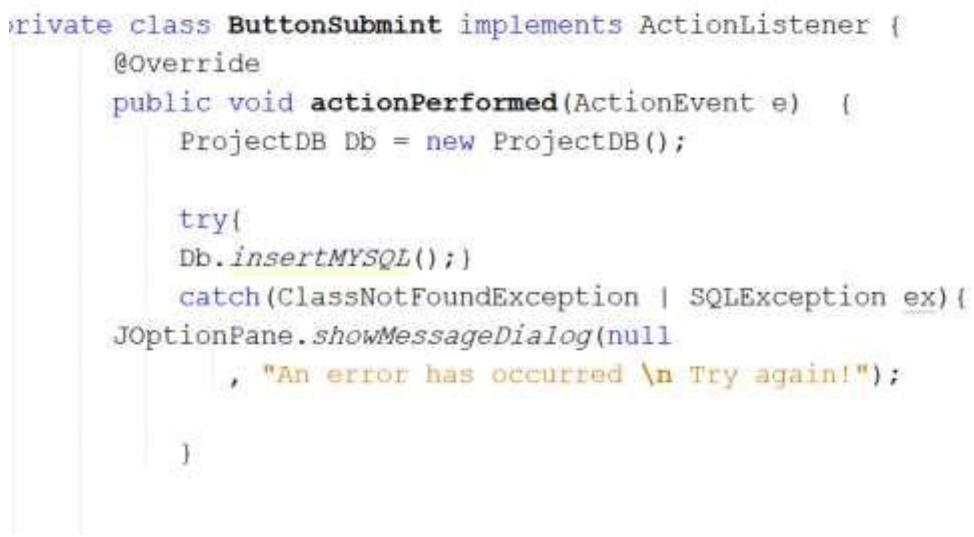
```
 postalcode= new JLabel ("Postal Code:",SwingConstants.LEFT ) ;
 postalcode.setFont(new Font ("Serif",Font.PLAIN ,20));
 postalcode_t= new JTextField (15);

phoneNumber = new JLabel ("Phone Number :",SwingConstants.LEFT )
phoneNumber.setFont(new Font ("Serif",Font.PLAIN ,20));
phoneNumber_t = new JTextField (15);

email = new JLabel ("Email:",SwingConstants.LEFT ) ;
email.setFont(new Font ("Serif",Font.PLAIN ,20));
email_t = new JTextField (15);

incom = new JLabel("Income :");
incom.setFont(new Font ("Serif",Font.PLAIN ,20));

panel_row1=  new JPanel();
panel_row1.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));
panel_row2=  new JPanel();
panel_row2.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));

panel_row1.add(Id);
panel_row1.add(Id_t);
panel_row1.add(name);
panel_row1.add(name_t);
panel_row1.add(DateBrith);
```



```
 panel_row2.add(incom_combo = new JComboBox(new String[]{"500","1000",
     "5000","10000","15000","20000","25000","30000" }));


all=new JPanel();
all.setLayout(new GridLayout(1,2,10,10));
all.add(panel_row1);
all.add(panel_row2);




add(all,BorderLayout.CENTER);


  // button
JButton Submint = new JButton("Submint");
JButton Cancel =new JButton("Cancel") ;

panel_row3=  new JPanel();
panel_row3.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));
panel_row3.add(Submint);
panel_row3.add(Cancel);
```



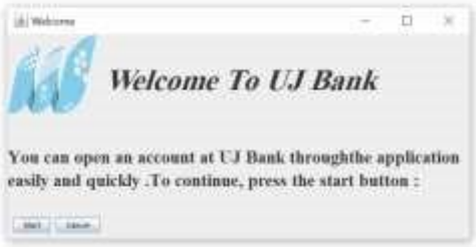
```
 }
    private class ButtonCo implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e)  {
          System.exit(0);

 }
 }
```



```
rivate class ButtonSubmint implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e)   {
             ProjectDB Db = new ProjectDB();

             try{
             Db.insertMYSQL();}
             catch(ClassNotFoundException | SQLException ex){
        JOptionPane.showMessageDialog(null
                  , "An error has occurred \n Try again!");

             }
```

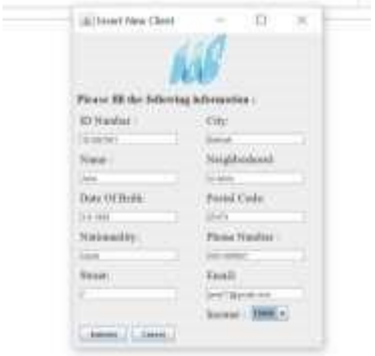
3.4

Interface


Welcome

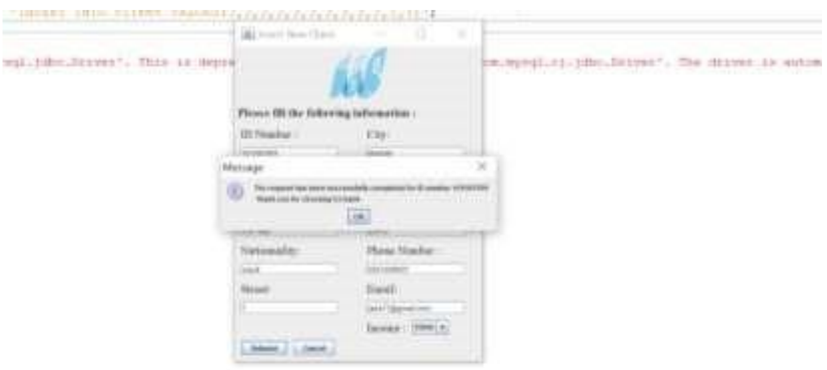
Welcome To UJ Bank

You can open an account at UJ Bank throughthe application easily and quickly .To continue, press the start button :


Insert New Client

Please fill the following information :

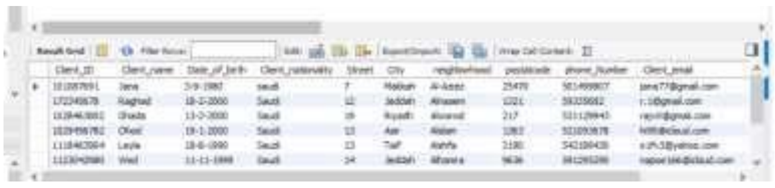
ID Number : [illegible]	City: [illegible]
Name : [illegible]	Neighborhood: [illegible]
Date Of Birth: [illegible]	Postal Code: [illegible]
Nationality: [illegible]	Phone Number : [illegible]
Street: [illegible]	Email: [illegible]
	Income : [illegible]


Insert New Client

Please fill the following information :

Message

[illegible]



Client_ID	Client_name	Date_of_birth	Client_nationality	Street	City	neighborhood	postalcode	phone_Number	Client_email
[illegible]	Jana	3-9-1987	saudi	7	Makkah	Al-Azeez	25470	[illegible]	jana77@gmail.com
[illegible]	Raghad	18-2-2000	Saudi	12	Jeddah	Alnaeem	1221	[illegible]	r.1@gmail.com
[illegible]	Ghada	13-2-2000	Saudi	19	Riyadh	Alwarod	217	[illegible]	[illegible]@gmail.com
[illegible]	[illegible]	19-1-2000	Saudi	13	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]@icloud.com
[illegible]	Leyla	18-6-1990	Saudi	13	Taif	[illegible]	2180	[illegible]	[illegible]@yahoo.com
[illegible]	Wed	11-11-1998	Saudi	14	Jeddah	Alhamra	9636	[illegible]	[illegible]@icloud.com