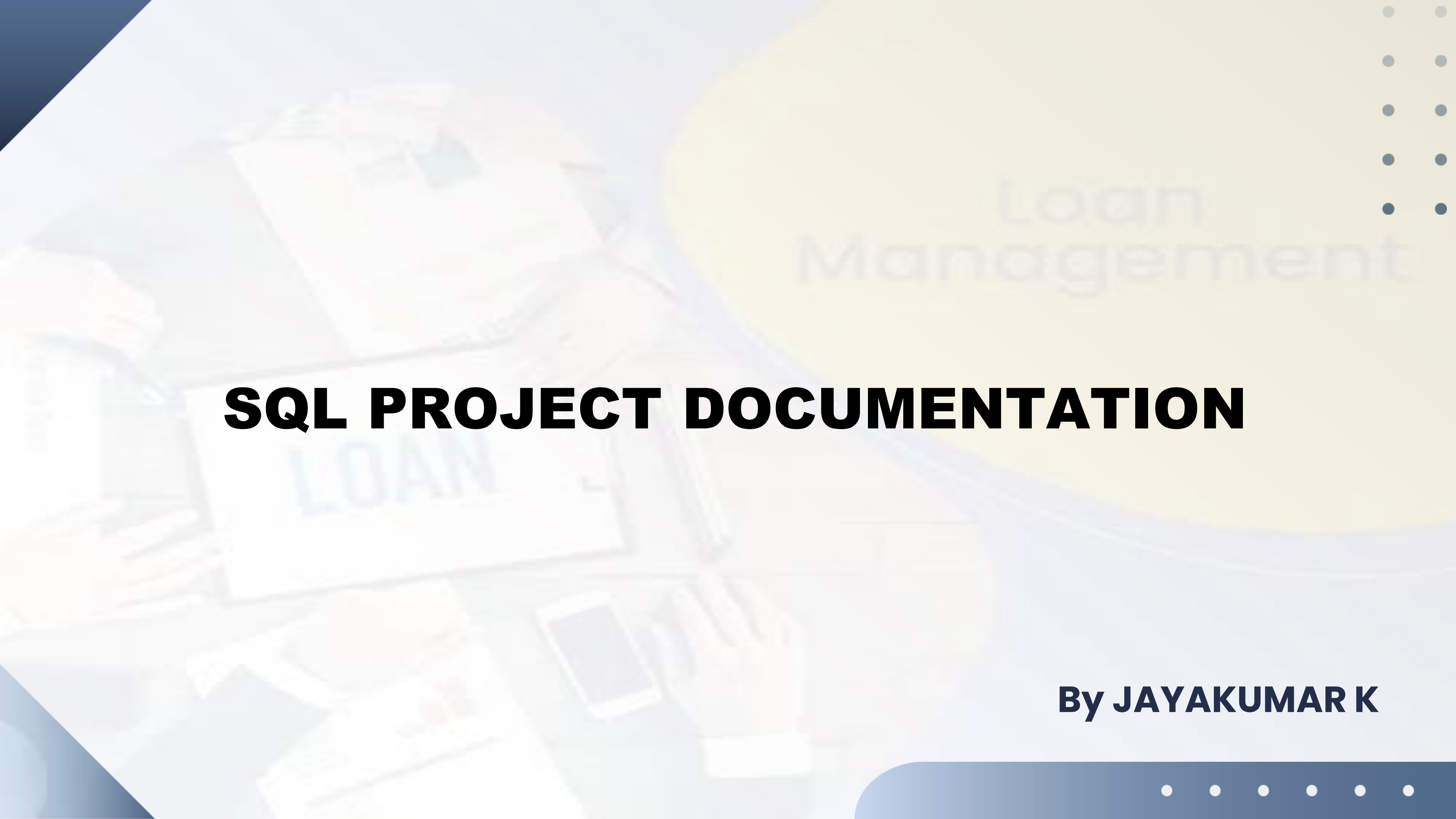


# SQL PROJECT DOCUMENTATION

By JAYAKUMAR K

# Introduction

SQL (Structured Query Language) is a standard language used to manage and manipulate relational databases. It allows users to create, read, update, and delete data efficiently.

## Key Features of SQL

- **Data Querying** – Retrieve data using SELECT with filtering, sorting, and grouping.
- **Data Manipulation** – Insert (INSERT), update (UPDATE), and delete (DELETE) records.
- **Data Definition** – Create, alter, and drop tables (CREATE, ALTER, DROP).
- **Joins & Relationships** – Combine data from multiple tables (INNER JOIN, LEFT JOIN, etc.).
- **Aggregation** – Perform calculations like SUM, COUNT, AVG for summaries.
- **Transactions** – Ensure data integrity with COMMIT, ROLLBACK, and ACID properties.
- **Indexing** – Improve query speed by creating indexes on columns.
- **Security & Access Control** – Manage permissions (GRANT, REVOKE) for user roles.

# History of SQL

- Developed in the 1970s by IBM researchers Donald D. Chamberlin and Raymond F. Boyce.
- Initially called SEQUEL (Structured English Query Language).
- Standardized by ANSI in 1986 and ISO in 1987.
- Over time, different versions and optimizations were introduced by database vendors.

## SQL Statements

DML-Data Manipulation Language

DQL –Data Query language

DCL –Data Control Language

TCL-Transaction Control Language

# Most Used Databases

| | |
|---|---|
| **MySQL**<br>Open-source, widely used for web applications. | **PostgreSQL**<br>Advanced open-source database with powerful features |
| **Microsoft SQL Server**<br>Developed by Microsoft, popular in businesses. | **Oracle Database**<br>Enterprise-level, known for performance and security |

# Data Types in SQL

- **Numeric** – INT, FLOAT, DECIMAL
- **Character** – CHAR, VARCHAR, TEXT
- **Date & Time** – DATE, TIME, DATETIME
- **Boolean** – TRUE/FALSE
- **Binary Data** – BLOB

# LOAN MANAGEMENT PROJECT

By  jayakumar k

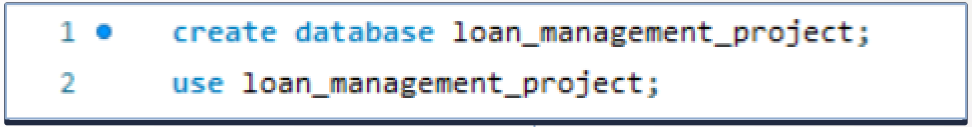
# overview

The **Loan Management System** analyzes customer income, loan status, and regional data for loan eligibility and risk assessment.

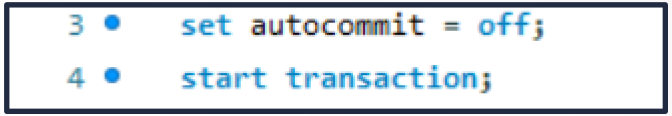
- ***customer income analysis****: categorizes income levels and calculates monthly & annual interest.*
- ***loan status & cibil score****: implements triggers to update loan processing and risk categories.*
- ***customer info update****: modifies gender and age details based on customer id.*
- ***geographical data integration****: merges country, state, and region details.*
- ***unified data output****: joins five tables to generate a consolidated dataset.*
- ***mismatch detection****: identifies inconsistencies in data joins.*
- ***data filtering****: extracts high cibil scores and customer types via inner joins.*
- ***stored procedures****: saves filtered outputs for efficient retrieval and reporting.*

# Creating Database

Creating database as “**Loan_management**” to use the database and after set the auto commit to ‘off’  to perform rollback and then start the transaction.



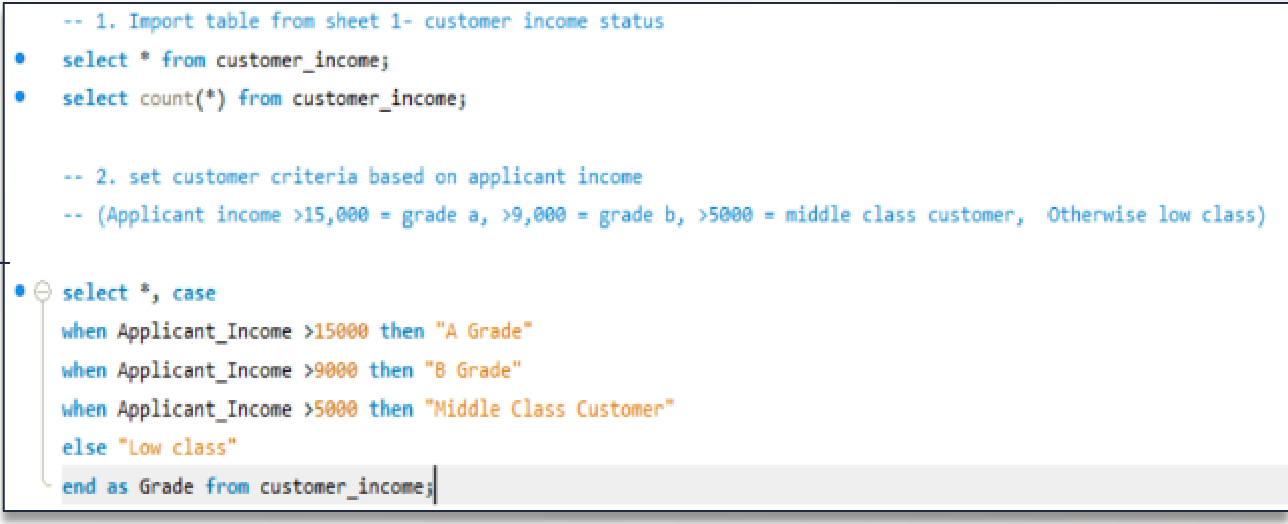
```sql
create database loan_management_project;
use loan_management_project;
```



```sql
set autocommit = off;
start transaction;
```

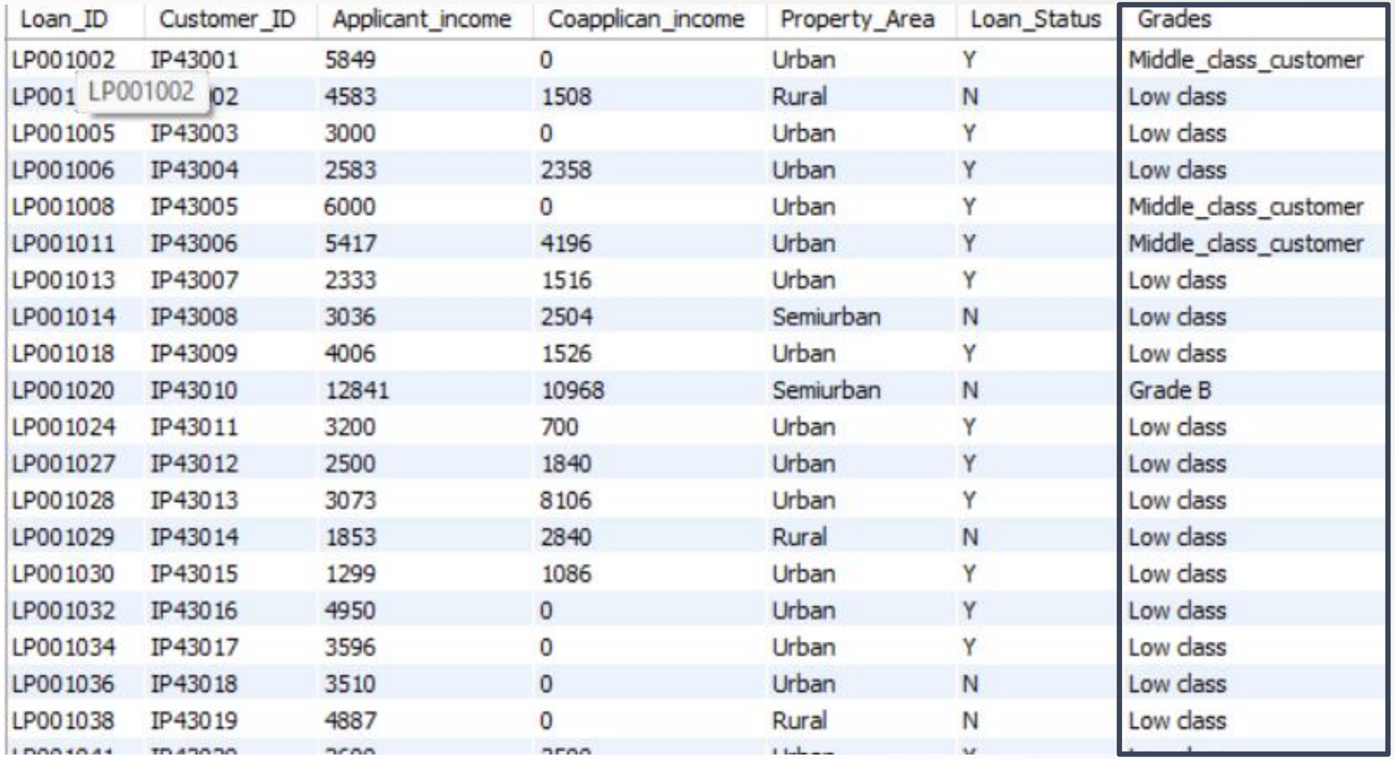
# Code to Set Criteria:

- Applicant income >15,000 = Grade A
- Applicant income >9,000 = Grade B
- Applicant income >5,000 = Middle class
- Applicant income <=5,000 = Low class



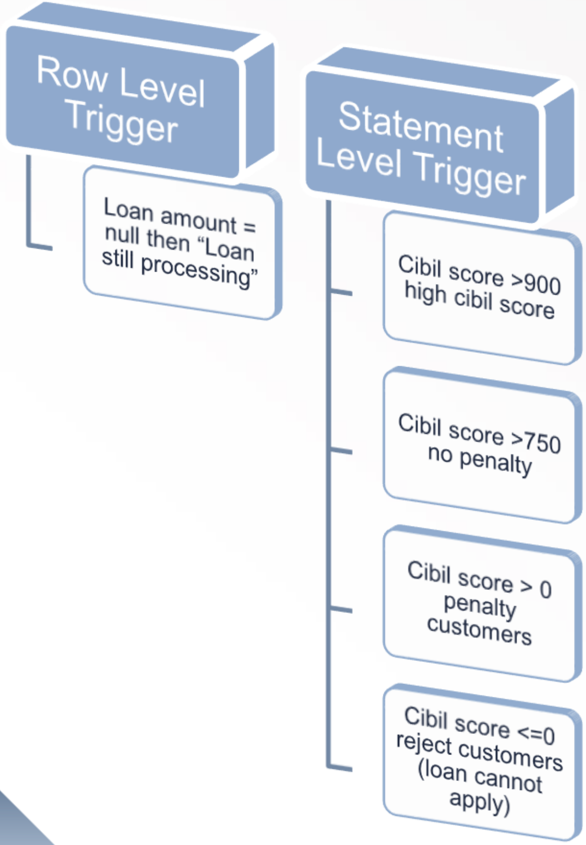
```
-- 1. Import table from sheet 1- customer income status
select * from customer_income;
select count(*) from customer_income;

-- 2. set customer criteria based on applicant income
-- (Applicant income >15,000 = grade a, >9,000 = grade b, >5000 = middle class customer,  Otherwise low class)

select *, case
when Applicant_Income >15000 then "A Grade"
when Applicant_Income >9000 then "B Grade"
when Applicant_Income >5000 then "Middle Class Customer"
else "Low class"
end as Grade from customer_income;
```

# OUTPUT



| Loan_ID | Customer_ID | Applicant_income | Coapplican_income | Property_Area | Loan_Status | Grades |
|---|---|---|---|---|---|---|
| LP001002 | IP43001 | 5849 | 0 | Urban | Y | Middle_class_customer |
| LP001 LP001002 | 02 | 4583 | 1508 | Rural | N | Low class |
| LP001005 | IP43003 | 3000 | 0 | Urban | Y | Low class |
| LP001006 | IP43004 | 2583 | 2358 | Urban | Y | Low class |
| LP001008 | IP43005 | 6000 | 0 | Urban | Y | Middle_class_customer |
| LP001011 | IP43006 | 5417 | 4196 | Urban | Y | Middle_class_customer |
| LP001013 | IP43007 | 2333 | 1516 | Urban | Y | Low class |
| LP001014 | IP43008 | 3036 | 2504 | Semiurban | N | Low class |
| LP001018 | IP43009 | 4006 | 1526 | Urban | Y | Low class |
| LP001020 | IP43010 | 12841 | 10968 | Semiurban | N | Grade B |
| LP001024 | IP43011 | 3200 | 700 | Urban | Y | Low class |
| LP001027 | IP43012 | 2500 | 1840 | Urban | Y | Low class |
| LP001028 | IP43013 | 3073 | 8106 | Urban | Y | Low class |
| LP001029 | IP43014 | 1853 | 2840 | Rural | N | Low class |
| LP001030 | IP43015 | 1299 | 1086 | Urban | Y | Low class |
| LP001032 | IP43016 | 4950 | 0 | Urban | Y | Low class |
| LP001034 | IP43017 | 3596 | 0 | Urban | Y | Low class |
| LP001036 | IP43018 | 3510 | 0 | Urban | N | Low class |
| LP001038 | IP43019 | 4887 | 0 | Rural | N | Low class |
| LP001041 | IP43020 | 2600 | 3500 | Urban | Y | [illegible] |

# Set Loan Status



Row Level Trigger

Loan amount = null then “Loan still processing”

Statement Level Trigger

Cibil score >900 high cibil score

Cibil score >750 no penalty

Cibil score > 0 penalty customers

Cibil score <=0 reject customers (loan cannot apply)

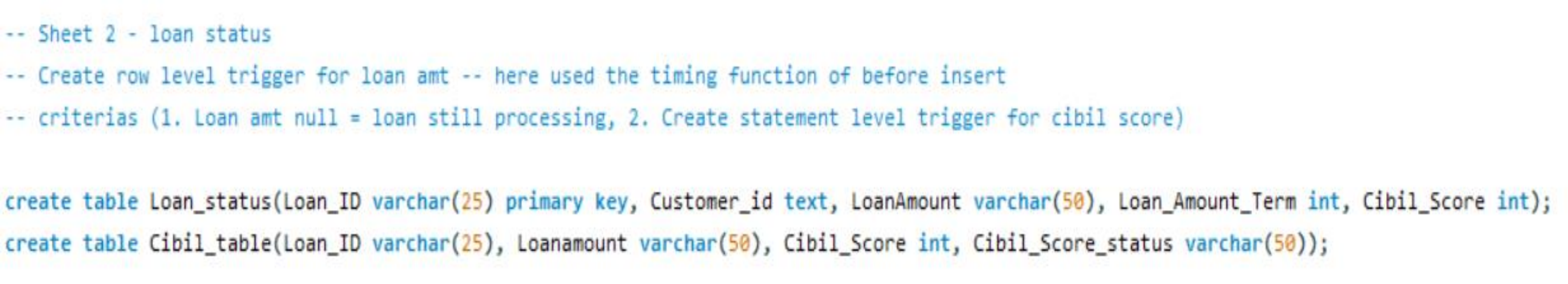
To set the value in the table, we need to create a trigger where it will check the specified condition. Once it satisfied, it will fire the value into the table.

Here we uses row level trigger and statement level trigger.

# Set Triggers

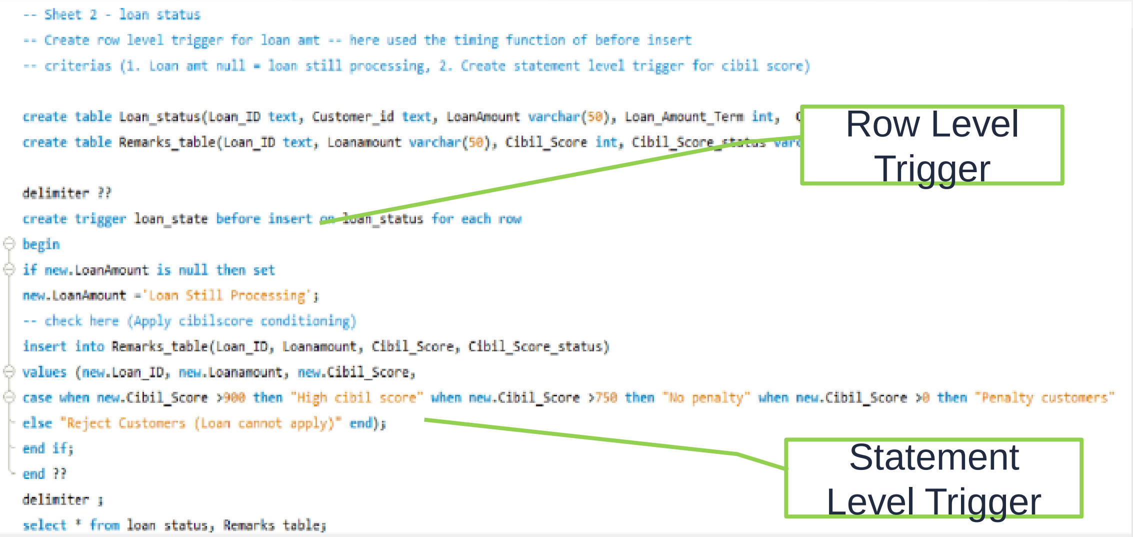
Create a table to get the status of loan and cibil score. Before insert the value into the table, initialize the “before insertion trigger” and set the criteria condition inside the trigger.

Once the value is inserted, trigger automatically get fired and then it check for condition depends on that it will change the value on the specified column.



```
-- Sheet 2 - loan status
-- Create row level trigger for loan amt -- here used the timing function of before insert
-- criterias (1. Loan amt null = loan still processing, 2. Create statement level trigger for cibil score)

create table Loan_status(Loan_ID varchar(25) primary key, Customer_id text, LoanAmount varchar(50), Loan_Amount_Term int, Cibil_Score int);
create table Cibil_table(Loan_ID varchar(25), Loanamount varchar(50), Cibil_Score int, Cibil_Score_status varchar(50));
```

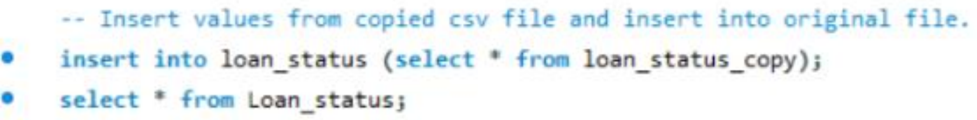
# Trigger Query’s & Output



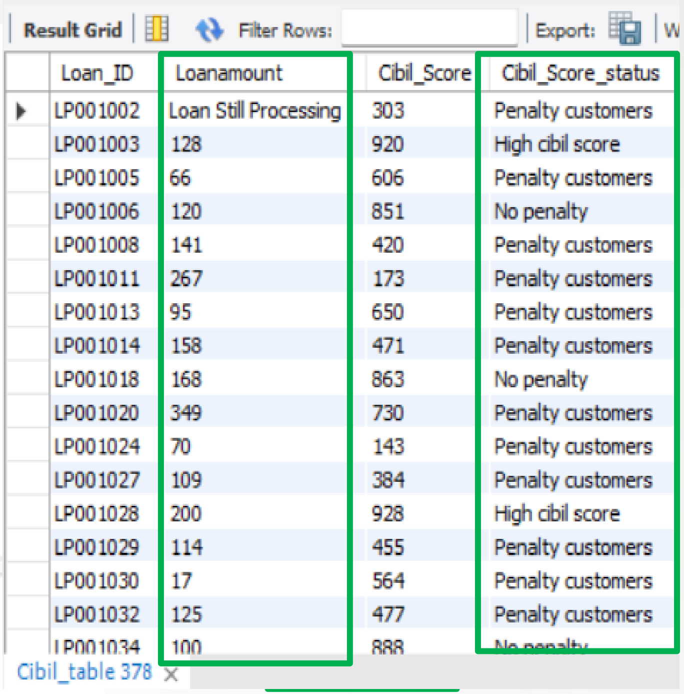
```
-- Sheet 2 - loan status
-- Create row level trigger for loan amt -- here used the timing function of before insert
-- criterias (1. Loan amt null = loan still processing, 2. Create statement level trigger for cibil score)

create table Loan_status(Loan_ID text, Customer_id text, LoanAmount varchar(50), Loan_Amount_Term int,  C
create table Remarks_table(Loan_ID text, Loanamount varchar(50), Cibil_Score int, Cibil_Score_status varc

delimiter ??
create trigger loan_state before insert on loan_status for each row
begin
if new.LoanAmount is null then set
new.LoanAmount ='Loan Still Processing';
-- check here (Apply cibilscore conditioning)
insert into Remarks_table(Loan_ID, Loanamount, Cibil_Score, Cibil_Score_status)
values (new.Loan_ID, new.Loanamount, new.Cibil_Score,
case when new.Cibil_Score >900 then "High cibil score" when new.Cibil_Score >750 then "No penalty" when new.Cibil_Score >0 then "Penalty customers"
else "Reject Customers (Loan cannot apply)" end);
end if;
end ??
delimiter ;
select * from loan_status, Remarks_table;
```

Row Level Trigger

Statement Level Trigger

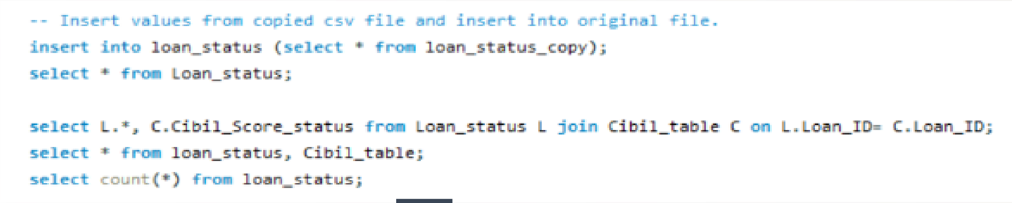


```
-- Insert values from copied csv file and insert into original file.
insert into loan_status (select * from loan_status_copy);
select * from Loan_status;
```

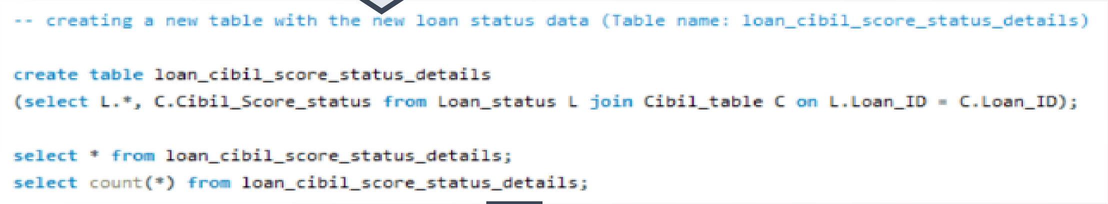


| Loan_ID | Loanamount | Cibil_Score | Cibil_Score_status |
|---|---|---|---|
| LP001002 | Loan Still Processing | 303 | Penalty customers |
| LP001003 | 128 | 920 | High cibil score |
| LP001005 | 66 | 606 | Penalty customers |
| LP001006 | 120 | 851 | No penalty |
| LP001008 | 141 | 420 | Penalty customers |
| LP001011 | 267 | 173 | Penalty customers |
| LP001013 | 95 | 650 | Penalty customers |
| LP001014 | 158 | 471 | Penalty customers |
| LP001018 | 168 | 863 | No penalty |
| LP001020 | 349 | 730 | Penalty customers |
| LP001024 | 70 | 143 | Penalty customers |
| LP001027 | 109 | 384 | Penalty customers |
| LP001028 | 200 | 928 | High cibil score |
| LP001029 | 114 | 455 | Penalty customers |
| LP001030 | 17 | 564 | Penalty customers |
| LP001032 | 125 | 477 | Penalty customers |
| LP001034 | 100 | 888 | No penalty |

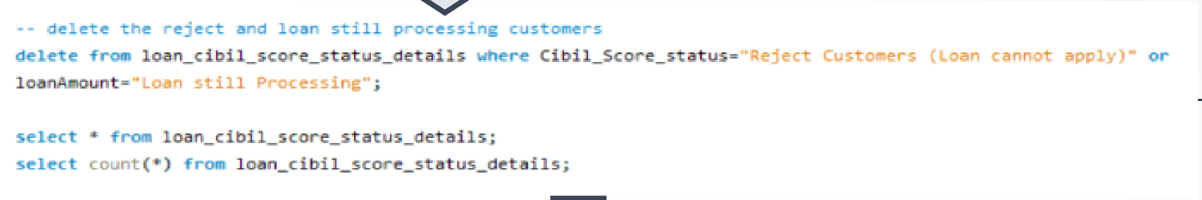
After execution and insertion, we join two tables and delete the loan “rejected” and “loan still processing” customers, then modify the loan amount column data type and create a new table using loan status table



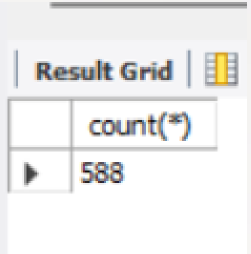
```
-- Insert values from copied csv file and insert into original file.
insert into loan_status (select * from loan_status_copy);
select * from Loan_status;

select L.*, C.Cibil_Score_status from Loan_status L join Cibil_table C on L.Loan_ID= C.Loan_ID;
select * from loan_status, Cibil_table;
select count(*) from loan_status;
```



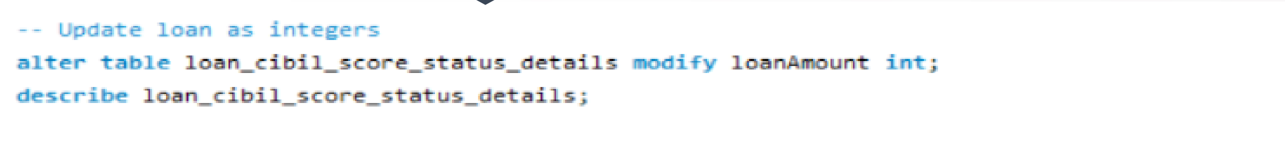
```
-- creating a new table with the new loan status data (Table name: loan_cibil_score_status_details)

create table loan_cibil_score_status_details
(select L.*, C.Cibil_Score_status from Loan_status L join Cibil_table C on L.Loan_ID = C.Loan_ID);

select * from loan_cibil_score_status_details;
select count(*) from loan_cibil_score_status_details;
```



```
-- delete the reject and loan still processing customers
delete from loan_cibil_score_status_details where Cibil_Score_status="Reject Customers (Loan cannot apply)" or
loanAmount="Loan still Processing";

select * from loan_cibil_score_status_details;
select count(*) from loan_cibil_score_status_details;
```



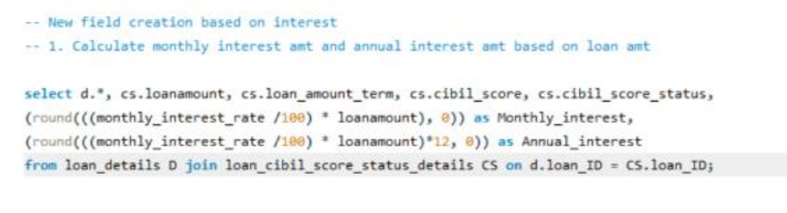
```
count(*)
588
```



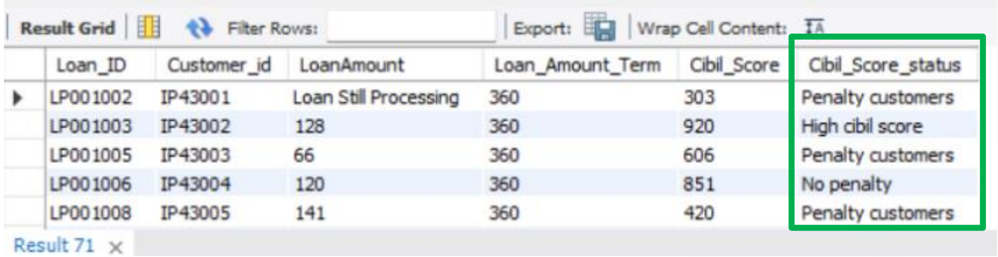
```
-- Update loan as integers
alter table loan_cibil_score_status_details modify loanAmount int;
describe loan_cibil_score_status_details;
```

# New field creation based on interest

Creating new field by calculating the monthly interest and annual interest based upon the loan amount and connect the customer income status and loan status by using “join” and form a new table.

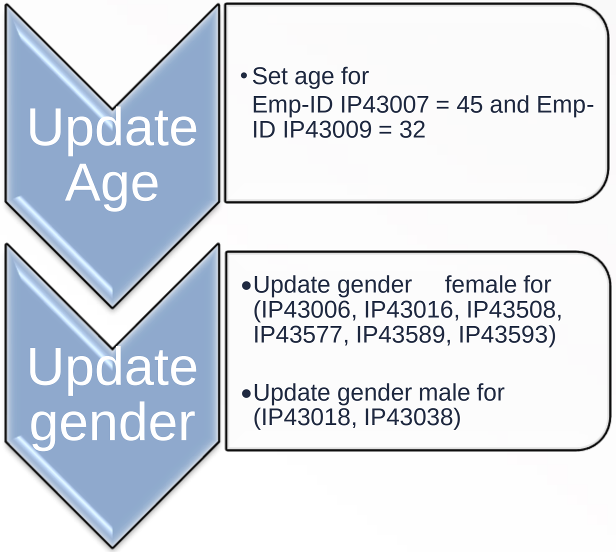


```sql
-- New field creation based on interest
-- 1. Calculate monthly interest amt and annual interest amt based on loan amt

select d.*, cs.loanamount, cs.loan_amount_term, cs.cibil_score, cs.cibil_score_status,
(round(((monthly_interest_rate /100) * loanamount), 0)) as Monthly_interest,
(round(((monthly_interest_rate /100) * loanamount)*12, 0)) as Annual_interest
from loan_details D join loan_cibil_score_status_details CS on d.loan_ID = CS.loan_ID;
```



| Loan_ID | Customer_id | LoanAmount | Loan_Amount_Term | Cibil_Score | Cibil_Score_status |
|---|---|---|---|---|---|
| LP001002 | IP43001 | Loan Still Processing | 360 | 303 | Penalty customers |
| LP001003 | IP43002 | 128 | 360 | 920 | High cibil score |
| LP001005 | IP43003 | 66 | 360 | 606 | Penalty customers |
| LP001006 | IP43004 | 120 | 360 | 851 | No penalty |
| LP001008 | IP43005 | 141 | 360 | 420 | Penalty customers |

# Updating Customer Info



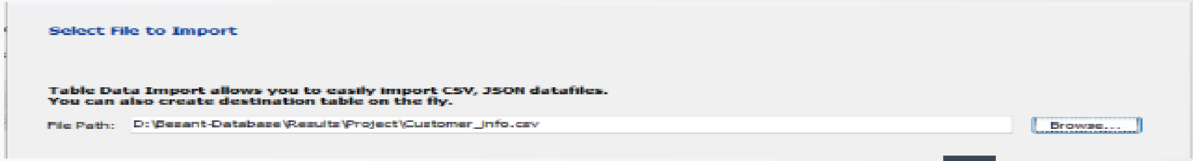
Update Age

- Set age for Emp-ID IP43007 = 45 and Emp-ID IP43009 = 32

Update gender

- Update gender female for (IP43006, IP43016, IP43508, IP43577, IP43589, IP43593)
- Update gender male for (IP43018, IP43038)

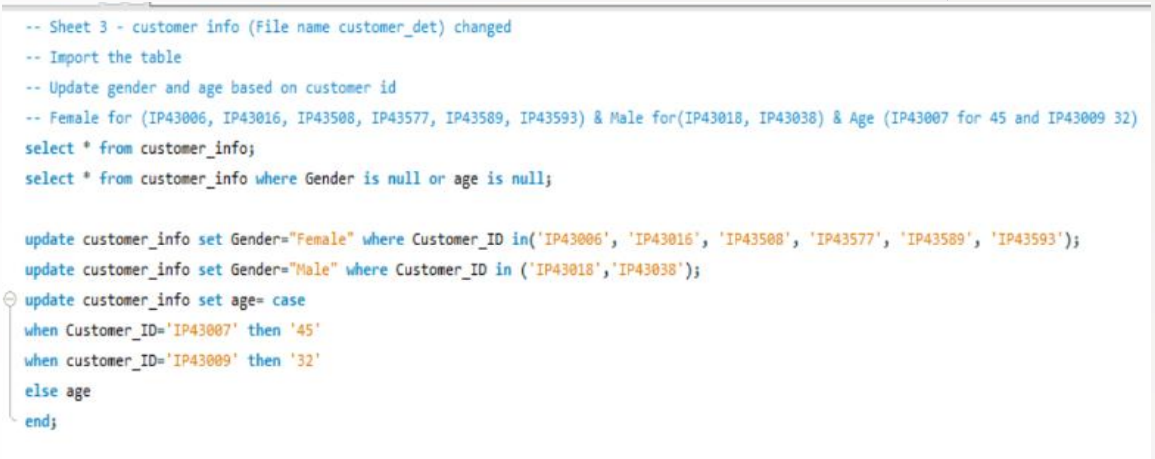
Here, Using Case end statement we update the specific customer gender and age.



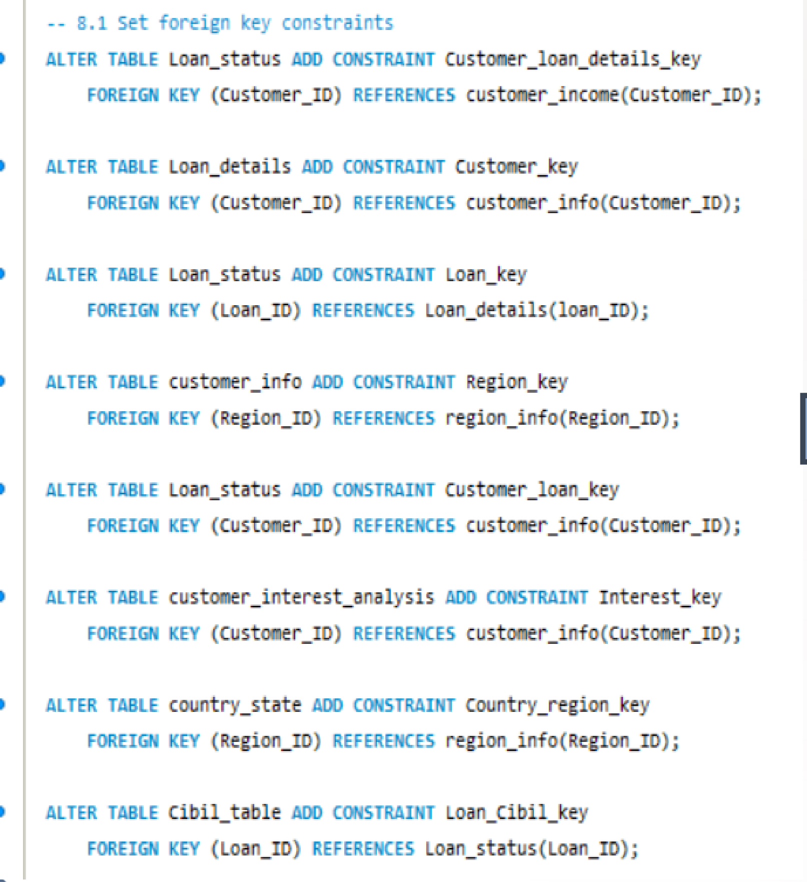
Select File to Import

Table Data Import allows you to easily import CSV, JSON datafiles.
You can also create destination table on the fly.

File Path: D:\Besant-Database\Results\Project\Customer_info.csv



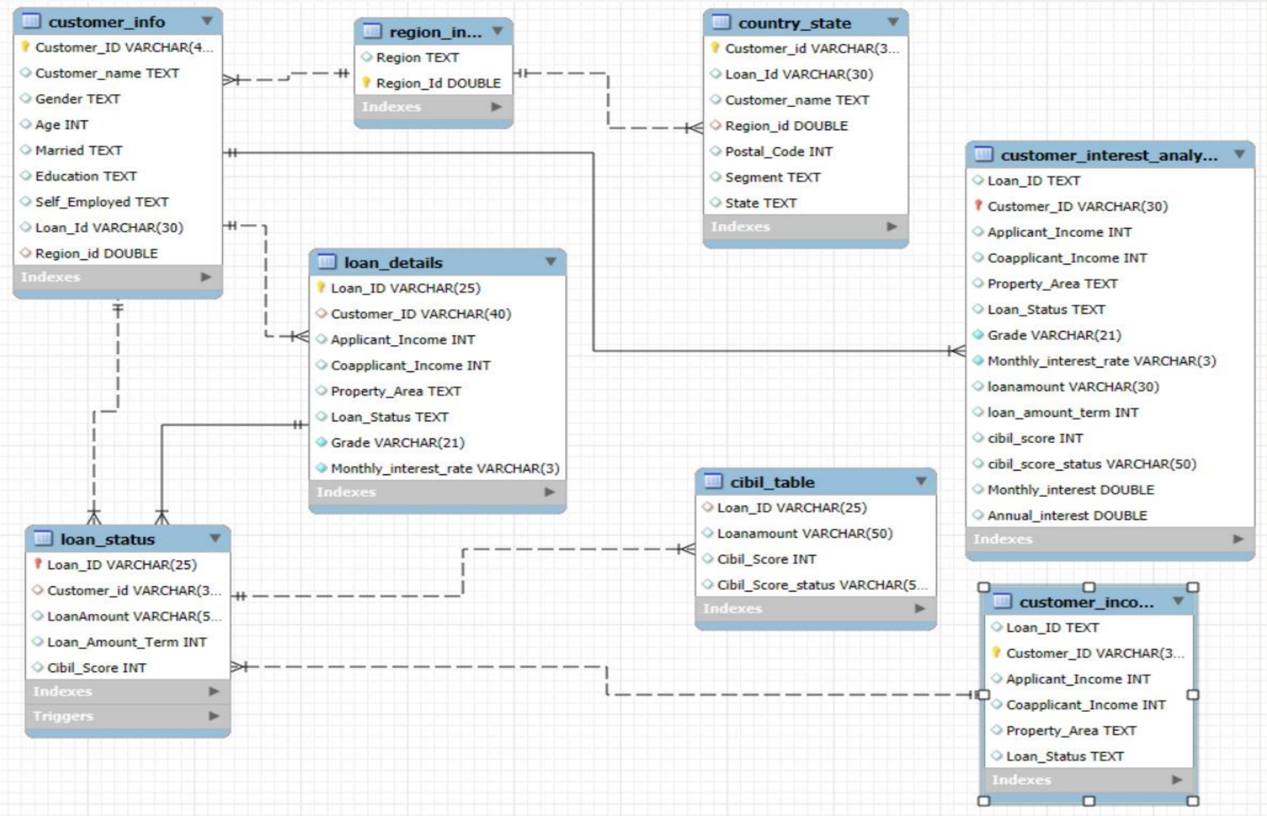
```
-- Sheet 3 - customer info (File name customer_det) changed
-- Import the table
-- Update gender and age based on customer id
-- Female for (IP43006, IP43016, IP43508, IP43577, IP43589, IP43593) & Male for(IP43018, IP43038) & Age (IP43007 for 45 and IP43009 32)
select * from customer_info;
select * from customer_info where Gender is null or age is null;

update customer_info set Gender="Female" where Customer_ID in('IP43006', 'IP43016', 'IP43508', 'IP43577', 'IP43589', 'IP43593');
update customer_info set Gender="Male" where Customer_ID in ('IP43018','IP43038');
update customer_info set age= case
when Customer_ID='IP43007' then '45'
when customer_ID='IP43009' then '32'
else age
end;
```



```sql
-- 8.1 Set foreign key constraints
ALTER TABLE Loan_status ADD CONSTRAINT Customer_loan_details_key
    FOREIGN KEY (Customer_ID) REFERENCES customer_income(Customer_ID);

ALTER TABLE Loan_details ADD CONSTRAINT Customer_key
    FOREIGN KEY (Customer_ID) REFERENCES customer_info(Customer_ID);

ALTER TABLE Loan_status ADD CONSTRAINT Loan_key
    FOREIGN KEY (Loan_ID) REFERENCES Loan_details(loan_ID);

ALTER TABLE customer_info ADD CONSTRAINT Region_key
    FOREIGN KEY (Region_ID) REFERENCES region_info(Region_ID);

ALTER TABLE Loan_status ADD CONSTRAINT Customer_loan_key
    FOREIGN KEY (Customer_ID) REFERENCES customer_info(Customer_ID);

ALTER TABLE customer_interest_analysis ADD CONSTRAINT Interest_key
    FOREIGN KEY (Customer_ID) REFERENCES customer_info(Customer_ID);

ALTER TABLE country_state ADD CONSTRAINT Country_region_key
    FOREIGN KEY (Region_ID) REFERENCES region_info(Region_ID);

ALTER TABLE Cibil_table ADD CONSTRAINT Loan_Cibil_key
    FOREIGN KEY (Loan_ID) REFERENCES Loan_status(Loan_ID);
```

# Adding Foreign key

# Fact dimension diagram



customer_info
- Customer_ID VARCHAR(4...
- Customer_name TEXT
- Gender TEXT
- Age INT
- Married TEXT
- Education TEXT
- Self_Employed TEXT
- Loan_Id VARCHAR(30)
- Region_id DOUBLE
- Indexes

region_in...
- Region TEXT
- Region_Id DOUBLE
- Indexes

country_state
- Customer_id VARCHAR(3...
- Loan_Id VARCHAR(30)
- Customer_name TEXT
- Region_id DOUBLE
- Postal_Code INT
- Segment TEXT
- State TEXT
- Indexes

customer_interest_analy...
- Loan_ID TEXT
- Customer_ID VARCHAR(30)
- Applicant_Income INT
- Coapplicant_Income INT
- Property_Area TEXT
- Loan_Status TEXT
- Grade VARCHAR(21)
- Monthly_interest_rate VARCHAR(3)
- loanamount VARCHAR(30)
- loan_amount_term INT
- cibil_score INT
- cibil_score_status VARCHAR(50)
- Monthly_interest DOUBLE
- Annual_interest DOUBLE
- Indexes

loan_details
- Loan_ID VARCHAR(25)
- Customer_ID VARCHAR(40)
- Applicant_Income INT
- Coapplicant_Income INT
- Property_Area TEXT
- Loan_Status TEXT
- Grade VARCHAR(21)
- Monthly_interest_rate VARCHAR(3)
- Indexes

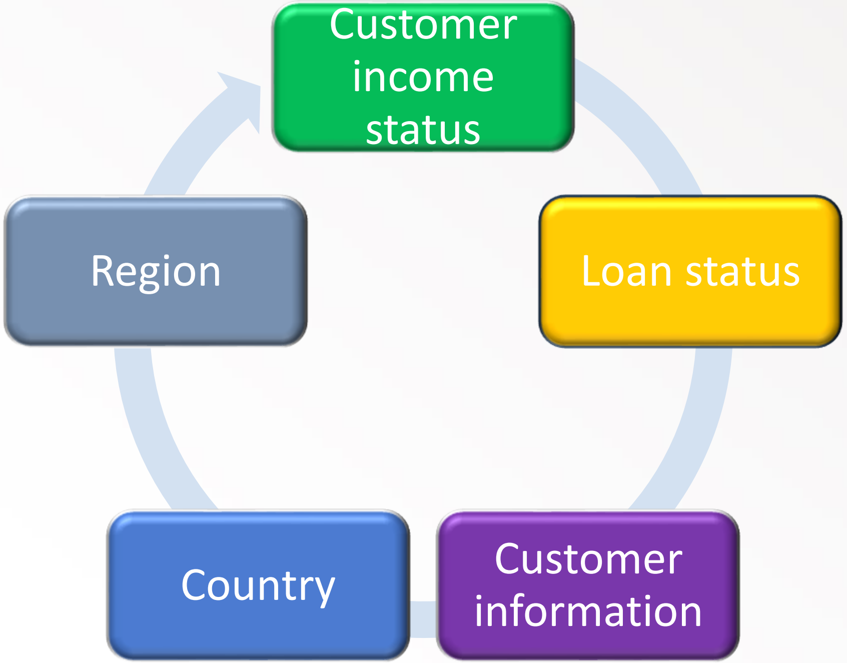
cibil_table
- Loan_ID VARCHAR(25)
- Loanamount VARCHAR(50)
- Cibil_Score INT
- Cibil_Score_status VARCHAR(5...
- Indexes

loan_status
- Loan_ID VARCHAR(25)
- Customer_id VARCHAR(3...
- LoanAmount VARCHAR(5...
- Loan_Amount_Term INT
- Cibil_Score INT
- Indexes
- Triggers

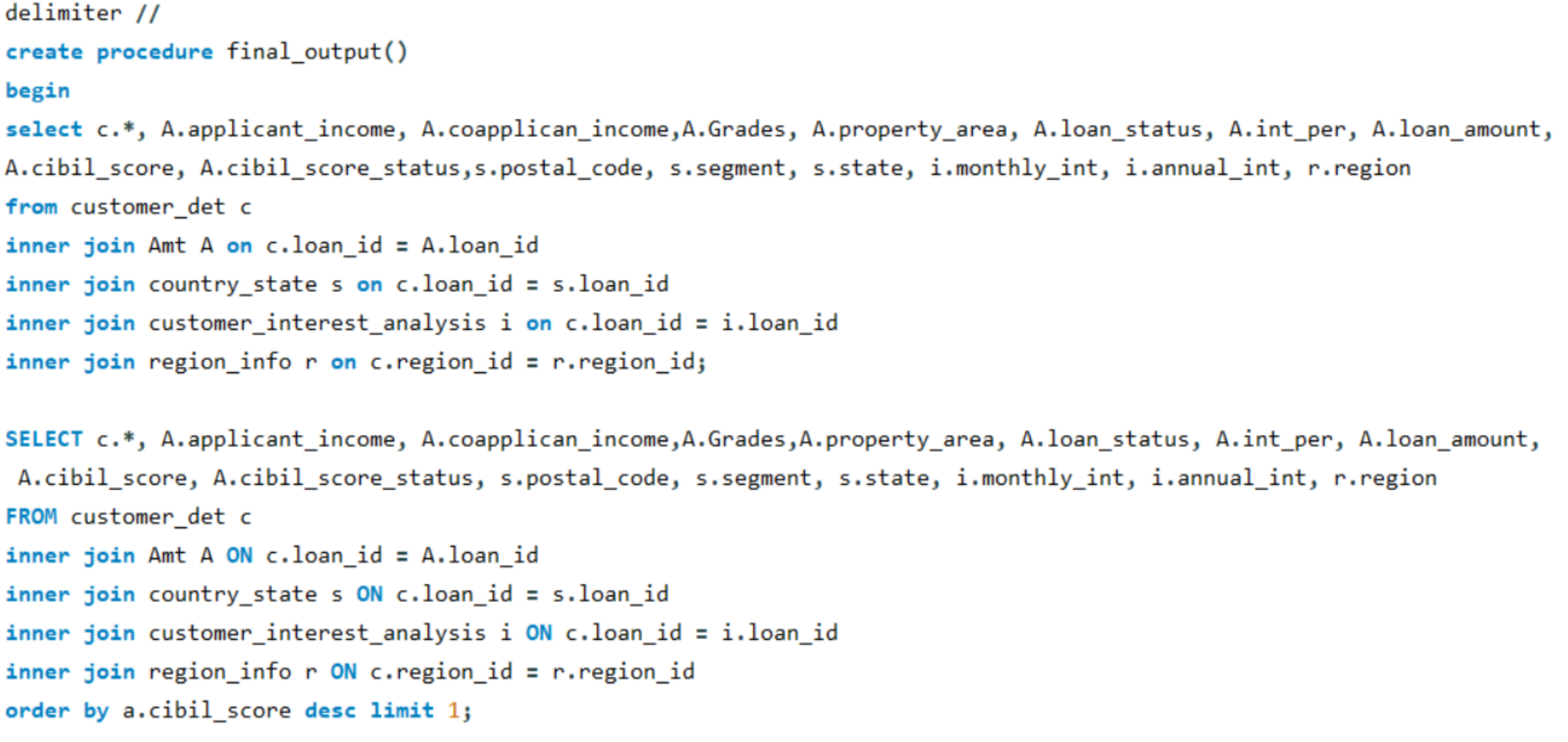
customer_inco...
- Loan_ID TEXT
- Customer_ID VARCHAR(3...
- Applicant_Income INT
- Coapplicant_Income INT
- Property_Area TEXT
- Loan_Status TEXT
- Indexes

# Joining Table

In this section, we import the customer’s country and region table and join all the tables which are related to customer’s info and loan details, then find the mismatch between each tables.

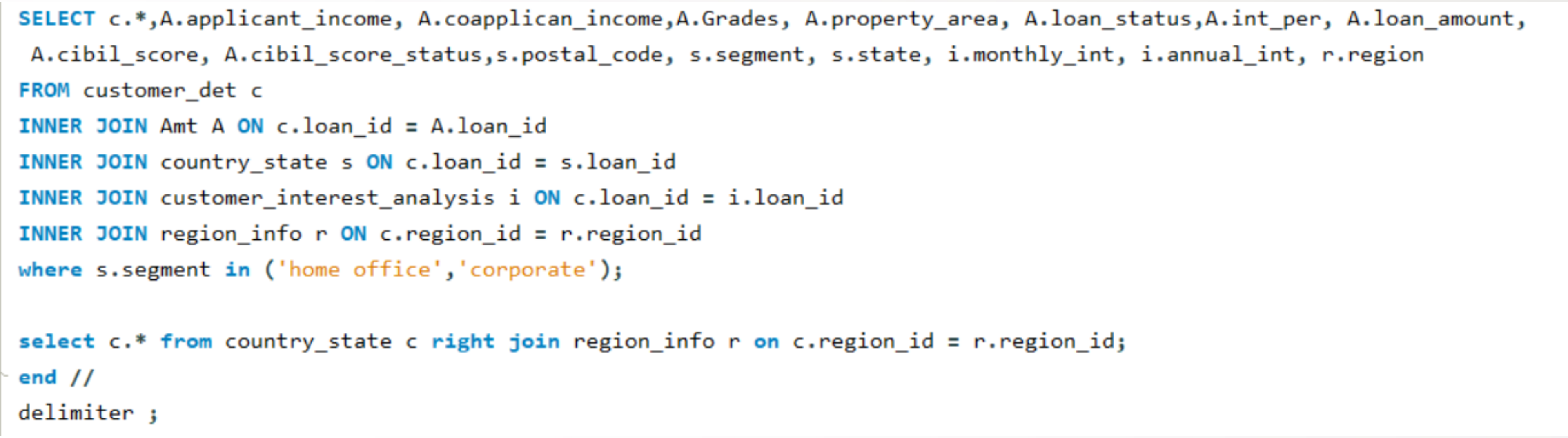


# Create a procedure called final_output that includes all join queries.



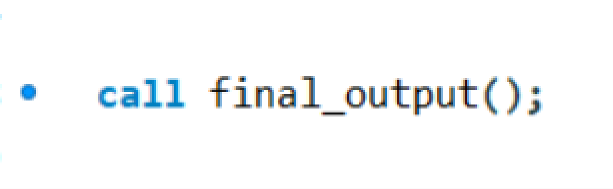
```
delimiter //
create procedure final_output()
begin
select c.*, A.applicant_income, A.coapplican_income,A.Grades, A.property_area, A.loan_status, A.int_per, A.loan_amount,
A.cibil_score, A.cibil_score_status,s.postal_code, s.segment, s.state, i.monthly_int, i.annual_int, r.region
from customer_det c
inner join Amt A on c.loan_id = A.loan_id
inner join country_state s on c.loan_id = s.loan_id
inner join customer_interest_analysis i on c.loan_id = i.loan_id
inner join region_info r on c.region_id = r.region_id;

SELECT c.*, A.applicant_income, A.coapplican_income,A.Grades,A.property_area, A.loan_status, A.int_per, A.loan_amount,
 A.cibil_score, A.cibil_score_status, s.postal_code, s.segment, s.state, i.monthly_int, i.annual_int, r.region
FROM customer_det c
inner join Amt A ON c.loan_id = A.loan_id
inner join country_state s ON c.loan_id = s.loan_id
inner join customer_interest_analysis i ON c.loan_id = i.loan_id
inner join region_info r ON c.region_id = r.region_id
order by a.cibil_score desc limit 1;
```

# Ending the procedure using delimiter



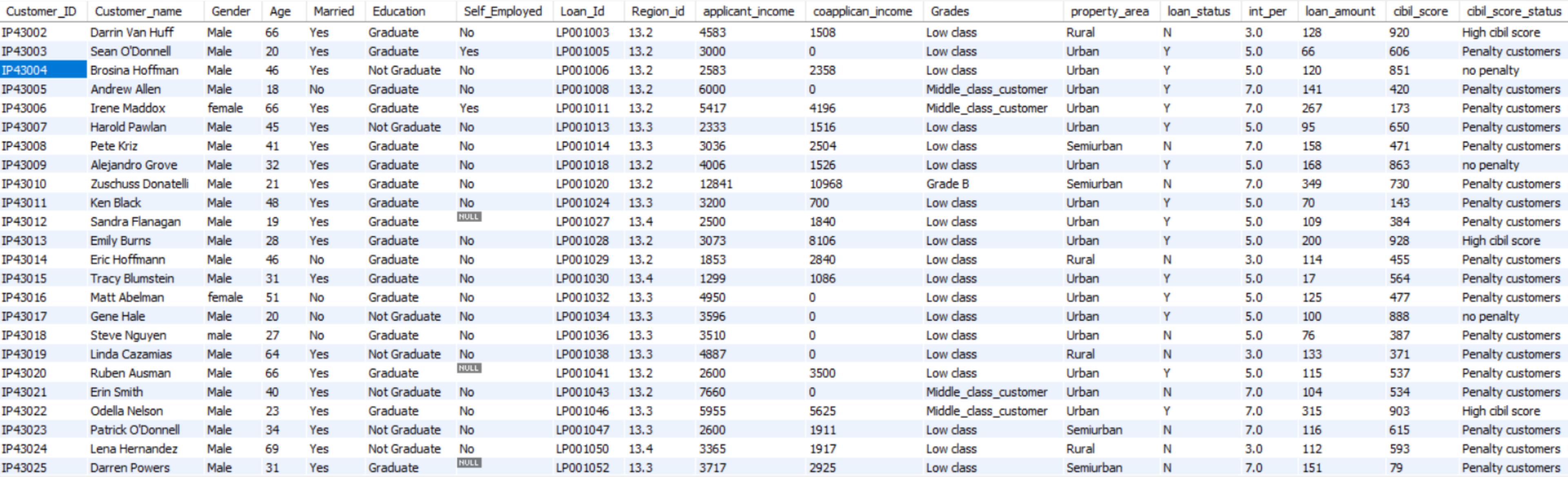
```
SELECT c.*,A.applicant_income, A.coapplican_income,A.Grades, A.property_area, A.loan_status,A.int_per, A.loan_amount,
 A.cibil_score, A.cibil_score_status,s.postal_code, s.segment, s.state, i.monthly_int, i.annual_int, r.region
FROM customer_det c
INNER JOIN Amt A ON c.loan_id = A.loan_id
INNER JOIN country_state s ON c.loan_id = s.loan_id
INNER JOIN customer_interest_analysis i ON c.loan_id = i.loan_id
INNER JOIN region_info r ON c.region_id = r.region_id
where s.segment in ('home office','corporate');

select c.* from country_state c right join region_info r on c.region_id = r.region_id;
end //
delimiter ;
```

# Calling procedure for all output



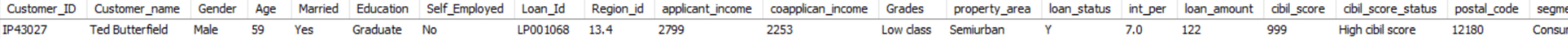
```
call final_output();
```

# Output 1



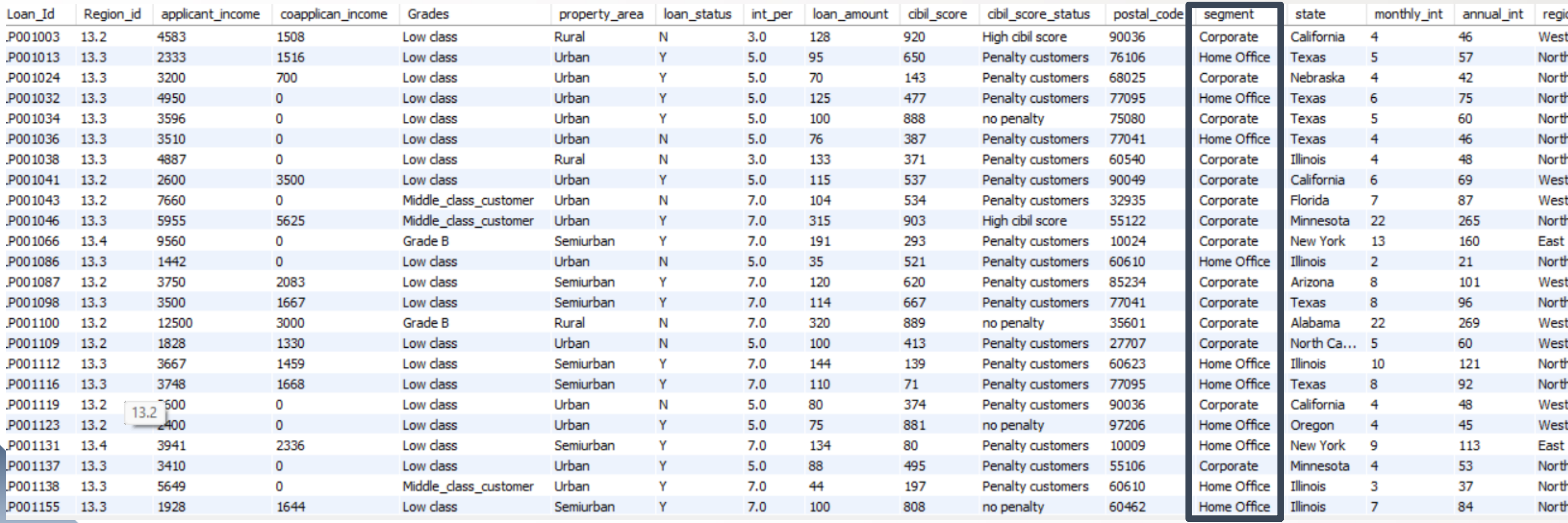
| Customer_ID | Customer_name | Gender | Age | Married | Education | Self_Employed | Loan_Id | Region_id | applicant_income | coapplican_income | Grades | property_area | loan_status | int_per | loan_amount | cibil_score | cibil_score_status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP43002 | Darrin Van Huff | Male | 66 | Yes | Graduate | No | LP001003 | 13.2 | 4583 | 1508 | Low class | Rural | N | 3.0 | 128 | 920 | High cibil score |
| IP43003 | Sean O'Donnell | Male | 20 | Yes | Graduate | Yes | LP001005 | 13.2 | 3000 | 0 | Low class | Urban | Y | 5.0 | 66 | 606 | Penalty customers |
| IP43004 | Brosina Hoffman | Male | 46 | Yes | Not Graduate | No | LP001006 | 13.2 | 2583 | 2358 | Low class | Urban | Y | 5.0 | 120 | 851 | no penalty |
| IP43005 | Andrew Allen | Male | 18 | No | Graduate | No | LP001008 | 13.2 | 6000 | 0 | Middle_class_customer | Urban | Y | 7.0 | 141 | 420 | Penalty customers |
| IP43006 | Irene Maddox | female | 66 | Yes | Graduate | Yes | LP001011 | 13.2 | 5417 | 4196 | Middle_class_customer | Urban | Y | 7.0 | 267 | 173 | Penalty customers |
| IP43007 | Harold Pawlan | Male | 45 | Yes | Not Graduate | No | LP001013 | 13.3 | 2333 | 1516 | Low class | Urban | Y | 5.0 | 95 | 650 | Penalty customers |
| IP43008 | Pete Kriz | Male | 41 | Yes | Graduate | No | LP001014 | 13.3 | 3036 | 2504 | Low class | Semiurban | N | 7.0 | 158 | 471 | Penalty customers |
| IP43009 | Alejandro Grove | Male | 32 | Yes | Graduate | No | LP001018 | 13.2 | 4006 | 1526 | Low class | Urban | Y | 5.0 | 168 | 863 | no penalty |
| IP43010 | Zuschuss Donatelli | Male | 21 | Yes | Graduate | No | LP001020 | 13.2 | 12841 | 10968 | Grade B | Semiurban | N | 7.0 | 349 | 730 | Penalty customers |
| IP43011 | Ken Black | Male | 48 | Yes | Graduate | No | LP001024 | 13.3 | 3200 | 700 | Low class | Urban | Y | 5.0 | 70 | 143 | Penalty customers |
| IP43012 | Sandra Flanagan | Male | 19 | Yes | Graduate | NULL | LP001027 | 13.4 | 2500 | 1840 | Low class | Urban | Y | 5.0 | 109 | 384 | Penalty customers |
| IP43013 | Emily Burns | Male | 28 | Yes | Graduate | No | LP001028 | 13.2 | 3073 | 8106 | Low class | Urban | Y | 5.0 | 200 | 928 | High cibil score |
| IP43014 | Eric Hoffmann | Male | 46 | No | Graduate | No | LP001029 | 13.2 | 1853 | 2840 | Low class | Rural | N | 3.0 | 114 | 455 | Penalty customers |
| IP43015 | Tracy Blumstein | Male | 31 | Yes | Graduate | No | LP001030 | 13.4 | 1299 | 1086 | Low class | Urban | Y | 5.0 | 17 | 564 | Penalty customers |
| IP43016 | Matt Abelman | female | 51 | No | Graduate | No | LP001032 | 13.3 | 4950 | 0 | Low class | Urban | Y | 5.0 | 125 | 477 | Penalty customers |
| IP43017 | Gene Hale | Male | 20 | No | Not Graduate | No | LP001034 | 13.3 | 3596 | 0 | Low class | Urban | Y | 5.0 | 100 | 888 | no penalty |
| IP43018 | Steve Nguyen | male | 27 | No | Graduate | No | LP001036 | 13.3 | 3510 | 0 | Low class | Urban | N | 5.0 | 76 | 387 | Penalty customers |
| IP43019 | Linda Cazamias | Male | 64 | Yes | Not Graduate | No | LP001038 | 13.3 | 4887 | 0 | Low class | Rural | N | 3.0 | 133 | 371 | Penalty customers |
| IP43020 | Ruben Ausman | Male | 66 | Yes | Graduate | NULL | LP001041 | 13.2 | 2600 | 3500 | Low class | Urban | Y | 5.0 | 115 | 537 | Penalty customers |
| IP43021 | Erin Smith | Male | 40 | Yes | Not Graduate | No | LP001043 | 13.2 | 7660 | 0 | Middle_class_customer | Urban | N | 7.0 | 104 | 534 | Penalty customers |
| IP43022 | Odella Nelson | Male | 23 | Yes | Graduate | No | LP001046 | 13.3 | 5955 | 5625 | Middle_class_customer | Urban | Y | 7.0 | 315 | 903 | High cibil score |
| IP43023 | Patrick O'Donnell | Male | 34 | Yes | Not Graduate | No | LP001047 | 13.3 | 2600 | 1911 | Low class | Semiurban | N | 7.0 | 116 | 615 | Penalty customers |
| IP43024 | Lena Hernandez | Male | 69 | Yes | Not Graduate | No | LP001050 | 13.4 | 3365 | 1917 | Low class | Rural | N | 3.0 | 112 | 593 | Penalty customers |
| IP43025 | Darren Powers | Male | 31 | Yes | Graduate | NULL | LP001052 | 13.3 | 3717 | 2925 | Low class | Semiurban | N | 7.0 | 151 | 79 | Penalty customers |

# Output 2



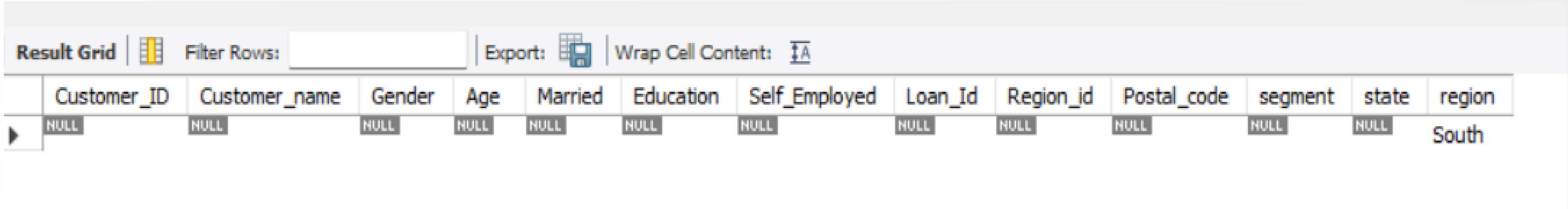
| Customer_ID | Customer_name | Gender | Age | Married | Education | Self_Employed | Loan_Id | Region_id | applicant_income | coapplican_income | Grades | property_area | loan_status | int_per | loan_amount | cibil_score | cibil_score_status | postal_code | segme |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP43027 | Ted Butterfield | Male | 59 | Yes | Graduate | No | LP001068 | 13.4 | 2799 | 2253 | Low class | Semiurban | Y | 7.0 | 122 | 999 | High cibil score | 12180 | Consu |

# Output 3



| Loan_Id | Region_id | applicant_income | coapplican_income | Grades | property_area | loan_status | int_per | loan_amount | cibil_score | cibil_score_status | postal_code | segment | state | monthly_int | annual_int | regi |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| .P001003 | 13.2 | 4583 | 1508 | Low class | Rural | N | 3.0 | 128 | 920 | High cibil score | 90036 | Corporate | California | 4 | 46 | West |
| .P001013 | 13.3 | 2333 | 1516 | Low class | Urban | Y | 5.0 | 95 | 650 | Penalty customers | 76106 | Home Office | Texas | 5 | 57 | North |
| .P001024 | 13.3 | 3200 | 700 | Low class | Urban | Y | 5.0 | 70 | 143 | Penalty customers | 68025 | Corporate | Nebraska | 4 | 42 | North |
| .P001032 | 13.3 | 4950 | 0 | Low class | Urban | Y | 5.0 | 125 | 477 | Penalty customers | 77095 | Home Office | Texas | 6 | 75 | North |
| .P001034 | 13.3 | 3596 | 0 | Low class | Urban | Y | 5.0 | 100 | 888 | no penalty | 75080 | Corporate | Texas | 5 | 60 | North |
| .P001036 | 13.3 | 3510 | 0 | Low class | Urban | N | 5.0 | 76 | 387 | Penalty customers | 77041 | Home Office | Texas | 4 | 46 | North |
| .P001038 | 13.3 | 4887 | 0 | Low class | Rural | N | 3.0 | 133 | 371 | Penalty customers | 60540 | Corporate | Illinois | 4 | 48 | North |
| .P001041 | 13.2 | 2600 | 3500 | Low class | Urban | Y | 5.0 | 115 | 537 | Penalty customers | 90049 | Corporate | California | 6 | 69 | West |
| .P001043 | 13.2 | 7660 | 0 | Middle_class_customer | Urban | N | 7.0 | 104 | 534 | Penalty customers | 32935 | Corporate | Florida | 7 | 87 | West |
| .P001046 | 13.3 | 5955 | 5625 | Middle_class_customer | Urban | Y | 7.0 | 315 | 903 | High cibil score | 55122 | Corporate | Minnesota | 22 | 265 | North |
| .P001066 | 13.4 | 9560 | 0 | Grade B | Semiurban | Y | 7.0 | 191 | 293 | Penalty customers | 10024 | Corporate | New York | 13 | 160 | East |
| .P001086 | 13.3 | 1442 | 0 | Low class | Urban | N | 5.0 | 35 | 521 | Penalty customers | 60610 | Home Office | Illinois | 2 | 21 | North |
| .P001087 | 13.2 | 3750 | 2083 | Low class | Semiurban | Y | 7.0 | 120 | 620 | Penalty customers | 85234 | Corporate | Arizona | 8 | 101 | West |
| .P001098 | 13.3 | 3500 | 1667 | Low class | Semiurban | Y | 7.0 | 114 | 667 | Penalty customers | 77041 | Corporate | Texas | 8 | 96 | North |
| .P001100 | 13.2 | 12500 | 3000 | Grade B | Rural | N | 7.0 | 320 | 889 | no penalty | 35601 | Corporate | Alabama | 22 | 269 | West |
| .P001109 | 13.2 | 1828 | 1330 | Low class | Urban | N | 5.0 | 100 | 413 | Penalty customers | 27707 | Corporate | North Ca... | 5 | 60 | West |
| .P001112 | 13.3 | 3667 | 1459 | Low class | Semiurban | Y | 7.0 | 144 | 139 | Penalty customers | 60623 | Home Office | Illinois | 10 | 121 | North |
| .P001116 | 13.3 | 3748 | 1668 | Low class | Semiurban | Y | 7.0 | 110 | 71 | Penalty customers | 77095 | Home Office | Texas | 8 | 92 | North |
| .P001119 | 13.2 | 2600 | 0 | Low class | Urban | N | 5.0 | 80 | 374 | Penalty customers | 90036 | Corporate | California | 4 | 48 | West |
| .P001123 | 13.2 | 2400 | 0 | Low class | Urban | Y | 5.0 | 75 | 881 | no penalty | 97206 | Home Office | Oregon | 4 | 45 | West |
| .P001131 | 13.4 | 3941 | 2336 | Low class | Semiurban | Y | 7.0 | 134 | 80 | Penalty customers | 10009 | Home Office | New York | 9 | 113 | East |
| .P001137 | 13.3 | 3410 | 0 | Low class | Urban | Y | 5.0 | 88 | 495 | Penalty customers | 55106 | Corporate | Minnesota | 4 | 53 | North |
| .P001138 | 13.3 | 5649 | 0 | Middle_class_customer | Urban | Y | 7.0 | 44 | 197 | Penalty customers | 60610 | Home Office | Illinois | 3 | 37 | North |
| .P001155 | 13.3 | 1928 | 1644 | Low class | Semiurban | Y | 7.0 | 100 | 808 | no penalty | 60462 | Home Office | Illinois | 7 | 84 | North |

# Output 4



| Customer_ID | Customer_name | Gender | Age | Married | Education | Self_Employed | Loan_Id | Region_id | Postal_code | segment | state | region |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | South |

Thankyou