**A Small Business** **FinTech Lender Database Model**

Team 16: Elisa Baldo, Ziwei Shi

January 31st, 2021

A hand holding a card

Description automatically generated with low confidence

1. **Flow of System**

The Database aims to help an independent FinTech Lender to store and report data efficiently of the lending operations to small businesses. The business user will be able to receive a variety of reports from the system that will help running the business more efficiently.

As a lending institution, the process starts with the client application. The user needs to assess **borrower eligibility** for the requested loan before being able to write it. It is important to register the **client general and financial information**. This is because once the client is registered, the user has it on the radar and even if the financial information provided is not fit to lend the loan requested, the employee can either inform the applicant on **which variable could improve** to get that specific loan or could offer a more suitable one. Once the client is approved, the funds are released.

The lender would want to check and evaluate the **borrower capability to pay the loan back and on time**. It is crucial for the lending process to receive payments back, and the business user needs to track down efficiently **missed instalments**.

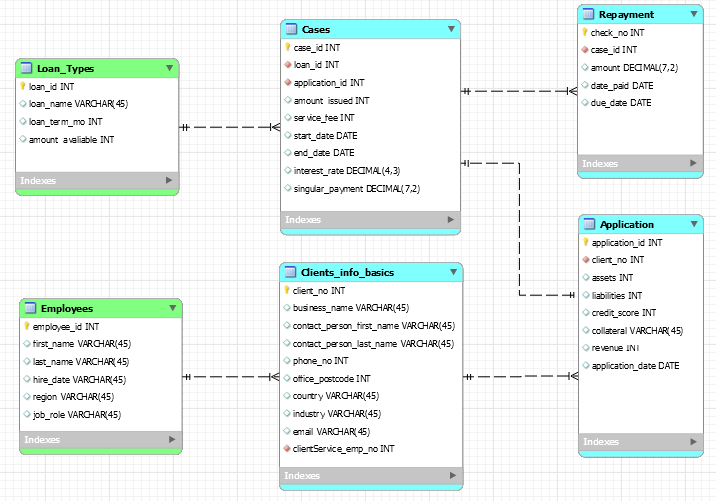
The FinTech would want to know which **agents and regions are performing best**. The managers would want to chase those agents that are having client with bad repayment habits or award the best performing.

Finally, the lender would want to **reach out to past clients** to offer new products and find new business.

1. **Database Structure**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ****Table Name**** | ****Primary Key(s)**** | ****Foreign Key(s)**** | ****Description**** |
| ****1**** | **Employees** | **employee\_id** | **N/A** | **To track the employees’ information** |
| ****2**** | **Clients\_info\_basics** | **client\_no** | **clientService\_**  **emp\_no** | **To record the basic info when customers registered** |
| ****3**** | **Loan Types** | **loan\_id** | **N/A** | **To list the type of loans available** |
| ****4**** | **Cases** | **case\_id** | **loan\_id,**  **application\_id** | **To track each business case operated between the company and customers** |
| ****5**** | **Application** | **application\_id** | **client\_no** | **Record application materials and info provided by clients** |
| ****6**** | **Repayment** | **check\_no** | **case\_id** | **Record all repayment activities made by customers** |

1. **Entity-Relational (ER) Model**



1. **SQL Queries**

**Query 1**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show those businesses that have registered to the lending platform but have not applied for any loan yet.** |
| ****Code**** | **select ci.client\_no, ci.business\_name as "Name"**  **from clients\_info\_basics ci**  **left join application a**  **on ci.client\_no = a.client\_no**  **where a.client\_no is null;** |
| ****Output**** |  |

**Query 2**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show those businesses that have had a loan and have finished repaying it and are now inactive, but their details are still on our record.** |
| ****Code**** | **select ci.client\_no, ci.business\_name as "Name",**  **concat\_ws(" ", ci.contact\_person\_first\_name,**  **ci.contact\_person\_last\_name) as "Contact Person" ,**  **ci.phone\_no, ci.email,**  **c.end\_date as "Case End Date"**  **from clients\_info\_basics ci**  **right join application a**  **on ci.client\_no = a.client\_no**  **right join cases c**  **on a.application\_id = c.application\_id**  **where end\_date < curdate();** |
| ****Output**** |  |

**Query 3**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show businesses that have been granted a loan and repaid on time and the right amounts.** |
| ****Code**** | **select distinct(ci.client\_no), ci.business\_name as "Name"**  **from clients\_info\_basics ci**  **right join application a**  **on ci.client\_no = a.client\_no**  **right join cases c**  **on a.application\_id = c.application\_id**  **right join repayment rp**  **on c.case\_id = rp.case\_id**  **where rp.date\_paid <= rp.due\_date**  **and rp.amount >= c.singular\_payment;** |
| ****Output**** |  |

**Query 4**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show businesses that have been granted a loan, but they have overdue repayments, or they have repaid not the right amount.** |
| ****Code**** | **select distinct(ci.client\_no), ci.business\_name as "Name"**  **from clients\_info\_basics ci**  **right join application a**  **on ci.client\_no = a.client\_no**  **right join cases c**  **on a.application\_id = c.application\_id**  **right join repayment rp**  **on c.case\_id = rp.case\_id**  **where rp.date\_paid > rp.due\_date**  **or rp.amount < c.singular\_payment;** |
| ****Output**** |  |

**Query 5**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show businesses that have applied for a loan, but they are not eligible. Financial statements reporting issues occurred, therefore they do not meet the standard to be granted the loan.** |
| ****Code**** | **select ci.client\_no, ci.business\_name as "Name", ap.credit\_score,**  **ap.collateral, revenue**  **from clients\_info\_basics ci, application ap**  **where ci.client\_no = ap.client\_no**  **and ap.liabilities > ap.assets;** |
| ****Output**** |  |

**Query 6**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show the employees that have performed the best in terms of number of sales.** |
| ****Code**** | **select e.employee\_id,**  **concat\_ws(" ", e.first\_name, e.last\_name) as "Name",**  **count(c.case\_id) as "Number of Cases"**  **from employees e**  **right join clients\_info\_basics ci**  **on e.employee\_id = ci.clientService\_emp\_no**  **right join application a**  **on ci.client\_no = a.client\_no**  **right join cases c**  **on a.application\_id = c.application\_id**  **group by e.employee\_id**  **order by 'number of cases'**  **Limit 1;** |
| ****Output**** |  |

**Query 7**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show the country where the most businesses that are granted loan come from. This is the best performing country in terms number of sales.** |
| ****Code**** | **select l.loan\_id, l.loan\_name**  **from loan\_types l**  **left join cases c**  **on l.loan\_id = c.loan\_id**  **where c.loan\_id is null;** |
| ****Output**** |  |

**Query 8**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show the type of loans that are not sold to any business.** |
| ****Code**** | **select ci.client\_no, ci.business\_name as "Name"**  **from clients\_info\_basics ci**  **left join application a**  **on ci.client\_no = a.client\_no**  **where a.client\_no is null;** |
| ****Output**** |  |

**Query 9**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show the industry where the most businesses are granted loans. This is the best performing industry in terms of number of sales.** |
| ****Code**** | **select count(c.case\_id) as "Number of Cases", ci.industry**  **from clients\_info\_basics ci**  **right join application ap**  **on ci.client\_no = ap.client\_no**  **right join cases c**  **on ap.application\_id = c.application\_id**  **group by industry**  **order by count(c.case\_id) desc;** |
| ****Output**** |  |

**Query 10**

|  |  |
| --- | --- |
| ****Functionality**** | **Creates a report to show the loan types preferred in the industry which has highest number business with loans granted.** |
| ****Code**** | **select count(c.case\_id) as "Number of Cases", lt.loan\_name**  **from loan\_types lt**  **right join cases c**  **on lt.loan\_id = c.loan\_id**  **right join application ap**  **on c.application\_id = ap.application\_id**  **right join clients\_info\_basics ci**  **on ap.client\_no = ci.client\_no**  **where ci.industry = "Printing"**  **group by lt.loan\_name;** |
| ****Output**** |  |

1. **SQL Procedures**

**Procedure 1**

The procedure in the box below is designed to show whether a specific case is closed or active by simply inputting the case id. The procedure will pull data from the end date column of the specific row and compare to the current date. Cases with end date any day before current date will be marked “Closed”, otherwise it will be marked “Active”. This can be used for account maintenance and client management.

CREATE DEFINER=`root`@`localhost` PROCEDURE `case\_status\_check`(IN in\_case\_id INT, OUT c\_status\_out varchar(10))

BEGIN

declare case\_check varchar(10);

select end\_date

into case\_check

from cases

where case\_id = in\_case\_id;

if case\_check < curdate()

then set c\_status\_out = "Closed";

else

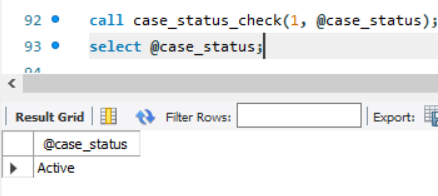
set c\_status\_out = "Active";

end if;

END

|  |  |  |
| --- | --- | --- |
| Test Case | Input | Output |
| 1 | 9 | Closed |
| 2 | 8 | Closed |
| 3 | 1 | Active |

Let’s check and see the procedure working with the example of case\_id = 1:



**Procedure 2**

The procedures in the boxes below, are created to pull information from the application table that reflect the eligibility of the client and give out a binary decision of “Approved” or “Denied”. The decision made by the procedure will be automatically updated to the decision column of application table. The standard we set here for “Approved” is revenue over 50,000 and assets over liabilities. Clients satisfy such standard will be approved and others will be denied. The drawback of the procedures is that the decision-making and filling process is realized by two single-loop procedures rather one multiple loop procedure. The result is only valid when both procedures are processed.

CREATE DEFINER=`root`@`localhost` PROCEDURE `eligibility\_check\_D`()

BEGIN

declare ap\_D INT;

declare D\_check cursor for

select application\_id

from application

where revenue < 50000

or assets < liabilities;

open D\_check;

app2:loop

fetch D\_check into ap\_D;

update application

set decision = "Denied"

where application\_id = ap\_D;

end loop app2;

END

CREATE DEFINER=`root`@`localhost` PROCEDURE `eligibility\_check\_A`()

BEGIN

declare ap\_A INT;

declare A\_check cursor for

select application\_id

from application

where revenue >= 50000

and assets >= liabilities;

open A\_check;

app1:loop

fetch A\_check into ap\_A;

update application

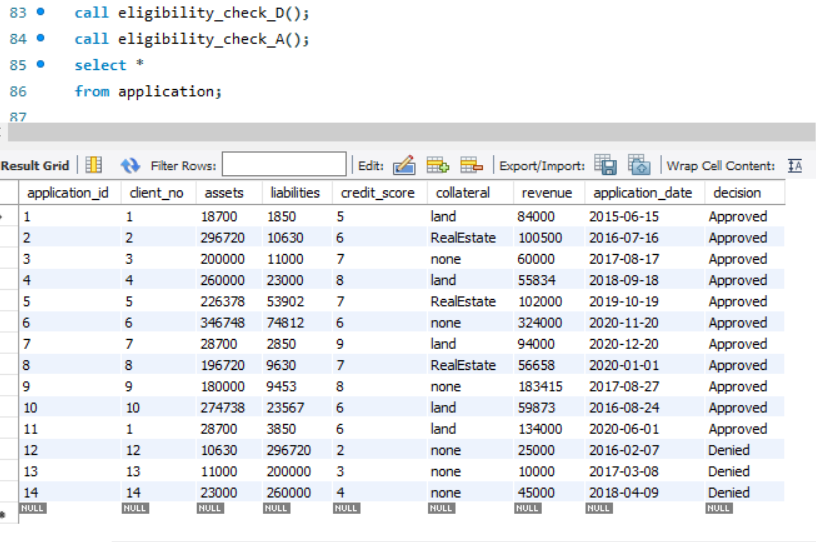
set decision = "Approved"

where application\_id = ap\_A;

end loop app1;

END

By calling the two procedures, the table is updated and by calling the table it is shown the status of each application.



1. **Appendix**

|  |  |
| --- | --- |
| Table Name | Script |
| Applications | CREATE TABLE `application` (  `application\_id` int NOT NULL,  `client\_no` int NOT NULL,  `assets` int DEFAULT NULL,  `liabilities` int DEFAULT NULL,  `credit\_score` int DEFAULT NULL,  `collateral` varchar(45) DEFAULT NULL,  `revenue` int DEFAULT NULL,  `application\_date` date DEFAULT NULL,  `decision` varchar(45) DEFAULT NULL,  PRIMARY KEY (`application\_id`),  KEY `fk\_Application\_Clients\_info\_basics1\_idx` (`client\_no`),  CONSTRAINT `fk\_Application\_Clients\_info\_basics1` FOREIGN KEY (`client\_no`) REFERENCES `clients\_info\_basics` (`client\_no`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |
| Cases | CREATE TABLE `cases` (  `case\_id` int NOT NULL,  `loan\_id` int NOT NULL,  `application\_id` int NOT NULL,  `amount\_issued` int DEFAULT NULL,  `service\_fee` int DEFAULT NULL,  `start\_date` date DEFAULT NULL,  `end\_date` date DEFAULT NULL,  `interest\_rate` decimal(4,3) DEFAULT NULL,  `singular\_payment` decimal(7,2) DEFAULT NULL,  `c\_status` varchar(45) DEFAULT NULL,  PRIMARY KEY (`case\_id`),  KEY `fk\_Case\_Details\_Loan\_Types1\_idx` (`loan\_id`),  KEY `fk\_Case\_Details\_Application1\_idx` (`application\_id`),  CONSTRAINT `fk\_Case\_Details\_Application1` FOREIGN KEY (`application\_id`) REFERENCES `application` (`application\_id`),  CONSTRAINT `fk\_Case\_Details\_Loan\_Types1` FOREIGN KEY (`loan\_id`) REFERENCES `loan\_types` (`loan\_id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |
| Clients\_info\_basics | CREATE TABLE `clients\_info\_basics` (  `client\_no` int NOT NULL,  `business\_name` varchar(45) DEFAULT NULL,  `contact\_person\_first\_name` varchar(45) DEFAULT NULL,  `contact\_person\_last\_name` varchar(45) DEFAULT NULL,  `phone\_no` int DEFAULT NULL,  `office\_postcode` int DEFAULT NULL,  `country` varchar(45) DEFAULT NULL,  `industry` varchar(45) DEFAULT NULL,  `email` varchar(45) DEFAULT NULL,  `clientService\_emp\_no` int NOT NULL,  PRIMARY KEY (`client\_no`),  KEY `fk\_Clients\_info\_basics\_Employees1\_idx` (`clientService\_emp\_no`),  CONSTRAINT `fk\_Clients\_info\_basics\_Employees1` FOREIGN KEY (`clientService\_emp\_no`) REFERENCES `employees` (`employee\_id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |
| Employees | CREATE TABLE `employees` (  `employee\_id` int NOT NULL,  `first\_name` varchar(45) DEFAULT NULL,  `last\_name` varchar(45) DEFAULT NULL,  `hire\_date` varchar(45) DEFAULT NULL,  `region` varchar(45) DEFAULT NULL,  `job\_role` varchar(45) DEFAULT NULL,  PRIMARY KEY (`employee\_id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |
| Loan\_types | CREATE TABLE `loan\_types` (  `loan\_id` int NOT NULL,  `loan\_name` varchar(45) DEFAULT NULL,  `loan\_term\_mo` int DEFAULT NULL,  `amount\_avaliable` int DEFAULT NULL,  PRIMARY KEY (`loan\_id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |
| Repayment | CREATE TABLE `repayment` (  `check\_no` int NOT NULL,  `case\_id` int NOT NULL,  `amount` decimal(7,2) DEFAULT NULL,  `date\_paid` date DEFAULT NULL,  `due\_date` date DEFAULT NULL,  PRIMARY KEY (`check\_no`),  KEY `fk\_Repayment\_Case1\_idx` (`case\_id`),  CONSTRAINT `fk\_Repayment\_Case1` FOREIGN KEY (`case\_id`) REFERENCES `cases` (`case\_id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |
|  | |