Data Modelling

Relational and Non-Relational

A part of GDDA707 Advanced Data Engineering Assessment-1

Prepared by: Aju Peter

**Table of Contents**

**Task-1 Data Modelling**

a. Application of Relational Data Modelling Techniques

b. Documentation of Data Model Structure

c. Normalisation

d. Application of Non-Relational Data Modelling Techniques

e. Documentation of Non-Relational Data Model Structure

**Task-2: Relational and Non-Relational Database Implementation [LO2]**

1. **A. Implement the designed data model into a relational database system**

a. Table Creation

b. Record Insertion

c. Update Queries

d. Duplicate records removal

e. Table Joins for Data Aggregation

1. **B. Implement a non-relational database model based on specific requirements or parts of the data model**

a. Collection Creation

b. Document Insertion

c. Update Queries

d. Duplicate records removal

e. Data Retrieval and Aggregation

**Documentation**

**Task 1: Data Modelling**

1. **Relational Data Modelling Techniques**

Scenario:

NZBank, a fictional bank with a few branches in New Zealand, has traditionally maintained its records in a conventional manner. The new manager, aiming to enhance the everyday banking operations, decided to hire a data engineer to develop a new database for their banking operations. As a data analyst, while reviewing the documents, I identified 5 key entities and their relationships (Table.1), and I created an ER model in Visio. This diagram (see Figure 1) serves as the blueprint for developing a relational database that supports the bank's objectives. By choosing to develop a new database, this bank is positioning itself to compete with modern banks while also contributing to the development of more efficient and secure banking services.

Below listed is the identified 5 key entities from the Bank record:

1. Bank: Details about each bank branch.
2. Customer: Information about customers.
3. Account: Customer accounts details.
4. Transaction: Account transactions (deposits and withdrawals).
5. Loan: Details of loans taken by customers.

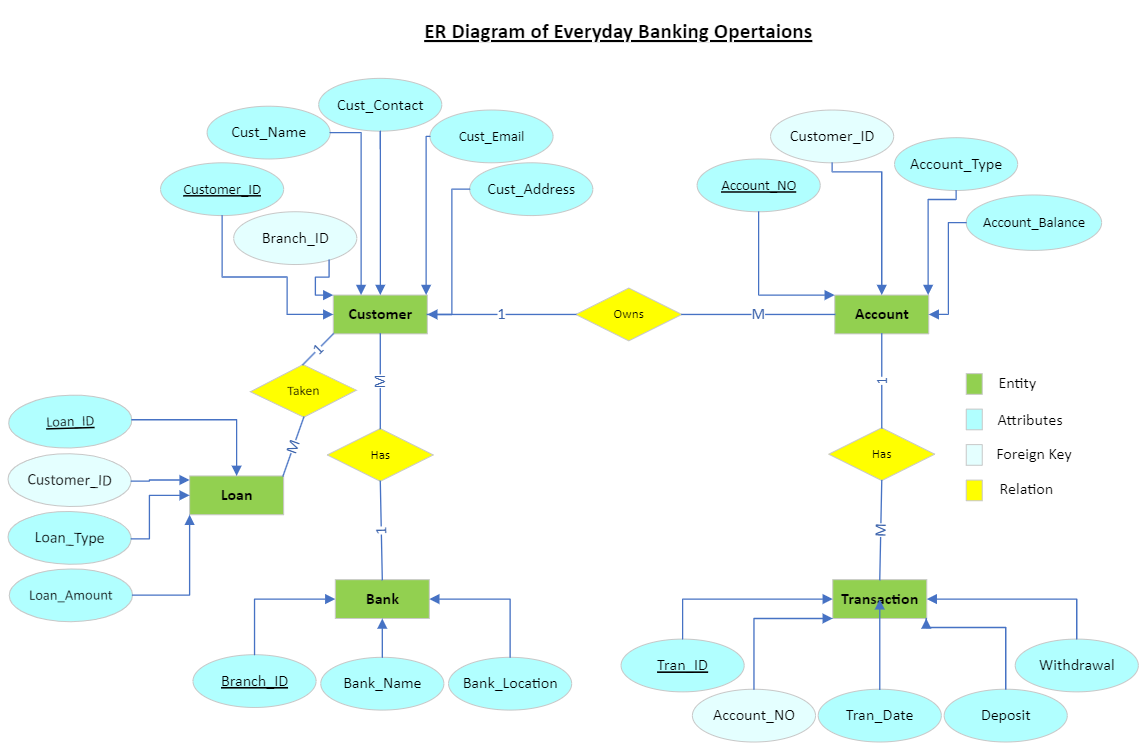


Figure 1.ER Diagram

Please find the ER diagram in Visio: [ER Diagram of Banking System.vsdx](https://nzschoolofed-my.sharepoint.com/:u:/g/personal/764706847_nzse_ac_nz/EfWBExD3yklPhabBppsQJykBPTrmVdNBqNEwTaE1AC2pFw?e=54RG4C)

Table 1.Entities and Attributes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bank | Customer | Account | Transaction | Loan |
| Branch\_ID (PK) | Customer\_ID (PK) | Account\_NO (PK) | Transaction\_ID (PK) | Loan\_ID (PK) |
| Bank\_Name | Branch\_ID (FK) | Customer\_ID (FK) | Account\_NO (FK) | Customer\_ID (FK) |
| Bank\_Contact | Customer\_Name | Account\_Type | Transaction\_Date | Loan\_Type |
| Bank\_Location | Customer\_Contact | Account\_Balance | Deposit | Loan\_Amount |
|  | Customer\_Email |  | Withdrawal |  |
|  | Customer\_Address |  |  |  |

1. **Documentation of Data Model Structure**

To document the data model structure for NZBank database comprehensively I have detailed the entities, attributes, and relationships based on the provided scenario. This documentation serves as a guide for understanding how the database is organized and how it can support NZBank's banking operations.

Below listed is the 5 key entities:

1. Bank: Details about each bank branch.
2. Customer: Information about customers.
3. Account: Customer accounts details.
4. Transaction: Account transactions (deposits and withdrawals).
5. Loan: Details of loans taken by customers.

Entities and Attributes:

Table 2.Entities and Attributes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bank | Customer | Account | Transaction | Loan |
| Branch\_ID (PK) | Customer\_ID (PK) | Account\_NO (PK) | Transaction\_ID (PK) | Loan\_ID (PK) |
| Bank\_Name | Branch\_ID (FK) | Customer\_ID (FK) | Account\_NO (FK) | Customer\_ID (FK) |
| Bank\_Contact | Customer\_Name | Account\_Type | Transaction\_Date | Loan\_Type |
| Bank\_Location | Customer\_Contact | Account\_Balance | Deposit | Loan\_Amount |
|  | Customer\_Email |  | Withdrawal |  |
|  | Customer\_Address |  |  |  |

1. Bank: Bank entity has multiple Branches and offers various accounts, loans, and facilitates of transactions to the customers.
2. Customer: A Customer is an individual holds one or more Accounts in the Bank. They may also have one or more Loans.
3. Account: Account belongs to a customer and is held at a Bank. Customers can have multiple Accounts.
4. Transaction: A Transaction is an operation that affects the balance of an Account. It could be a deposit, or withdrawal.
5. Loan: A Loan is borrowed by a Customer from a Bank. A Customer can have multiple Loans, but each Loan is associated with a single Customer and a single Bank.

Relationships: (please refer the Figure 1.ER Diagram)

1. Bank to Customer: One-to-Many - A bank branch can have many customers, but each customer belongs to one bank branch.
2. Customer to Account: One-to-Many - A customer can have multiple accounts, but each account belongs to one customer.
3. Account to Transaction: One-to-Many - An account can have multiple transactions, like; withdrawal and deposit.
4. Customer to Loan: One-to-Many - A customer can have multiple loans, but each loan is associated with one customer.

Primary Key (PK): Uniquely identifies each record within an entity.

Foreign Key (FK): Establishes a relationship between two entities, linking a record in one entity to a record in another.

This data model provides a detailed structure for NZBank's new database, enabling the bank to manage customer information, accounts, loans, and transactions efficiently. By documenting this model, we ensure that the database design is clear, scalable, and capable of supporting NZBank's to enhance operations in the modern banking landscape.

1. **Normalisation**

The Bank has provided the record in a consolidated CSV file format (Figure 2. Bank Raw Data). When I carefully analysed the data by using Python Software (Jupyter notebook), I found anomalies and redundancies. The initial phase of this analytical process involved a detailed identification of the key entities and their associated attributes, which is a critical step for structuring and normalizing the dataset.

To make an effective normalization procedure, I have used Python software. I uploaded the Bank data (CSV file) to the Jupyter notebook and imported necessary libraries like pandas and NumPy.

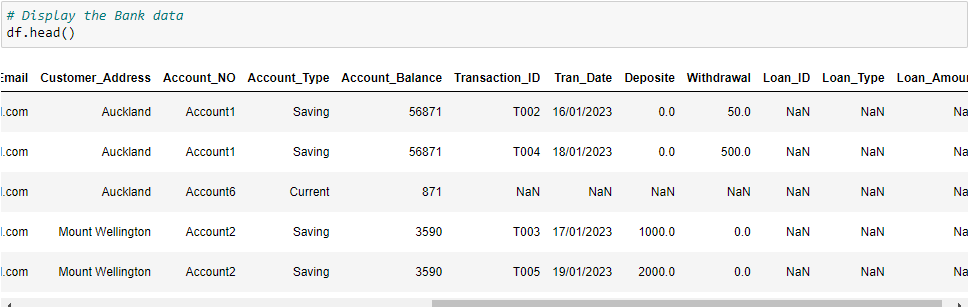


Figure 2.Bank Raw Data

Firstly, I split the data into the identified 5 key entities (Figure 3), then displayed each entity for further thorough analysis. During this process, special attention was dedicated to the identification and rectification of missing values and the elimination of repetitive records, which are often dangerous to the integrity and utility of the dataset.

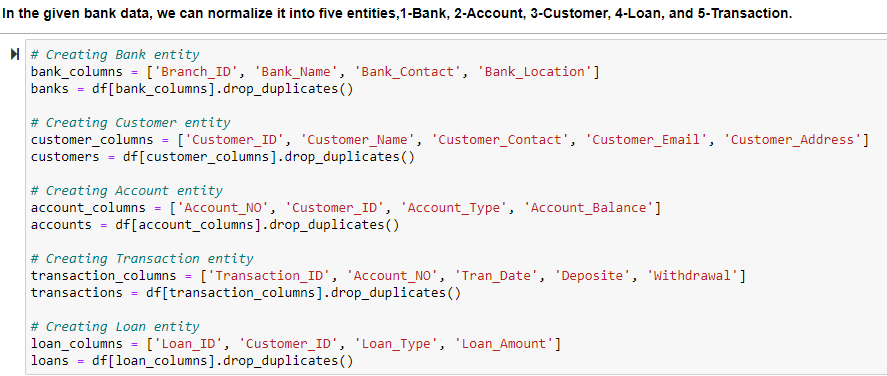


Figure 3.Splitting dataset

The subsequent removal of redundant data and records containing null values represented a crucial phase in the data normalization process. These measures ensured the dataset’s compliance with the First Normal Form (1NF) and the Second Normal Form (2NF).

1. Bank:

The Bank table (Figure 4) is in the first normal form (1NF), eliminated repeating groups such that all records in the table can be identified uniquely by a primary key (Branch\_ID) in the table. and all other attributes depend on the primary key.

It also meets the second normal form (2NF) as all non-key attributes fully depend on the primary key. Additionally, it might already be in the third normal form (3NF) if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

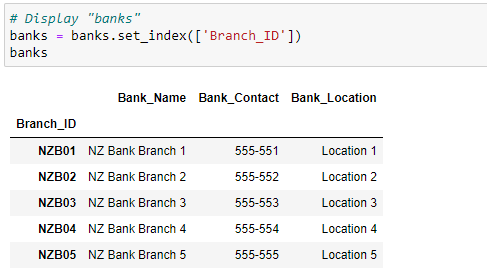


Figure 4.Bank table

1. Customers:

The Customer’s table (Figure 5) is in the first normal form (1NF), eliminated repeating groups such that all records in the table can be identified uniquely by a primary key (Customer\_ID) in the table. and all other attributes depend on the primary key.

It also meets the second normal form (2NF) as all non-key attributes fully depend on the primary key. Additionally, it might already be in the third normal form (3NF) if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

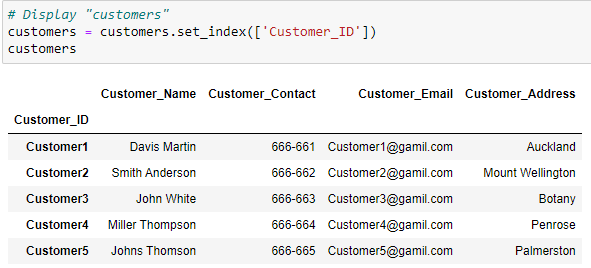


Figure 5.Customers table

1. Accounts:

The Accounts table (Figure 6) is in the first normal form (1NF), eliminated repeating groups such that all records in the table can be identified uniquely by a primary key (Account\_NO) in the table. and all other attributes depend on the primary key.

It also meets the second normal form (2NF) as all non-key attributes fully depend on the primary key. Additionally, it might already be in the third normal form (3NF) if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

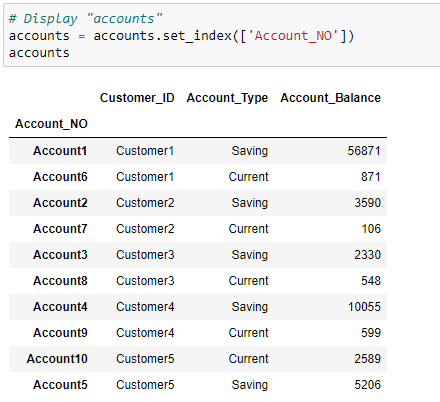


Figure 7.accounts table

1. Loans:

The Loan table (Figure 9) is in the first normal form (1NF), eliminated repeating groups such that all records in the table can be identified uniquely by a primary key (Loan\_ID) in the table. and all other attributes depend on the primary key.

It also meets the second normal form (2NF) as all non-key attributes fully depend on the primary key. Additionally, it might already be in the third normal form (3NF) if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

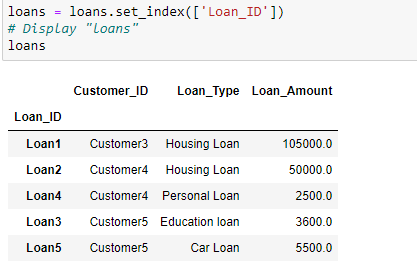


Figure 8.loan table

1. Transaction:

The Loan table (Figure 9) is in the first normal form (1NF), eliminated repeating groups such that all records in the table can be identified uniquely by a primary key (Transaction\_ID) in the table. and all other attributes depend on the primary key.

It also meets the second normal form (2NF) as all non-key attributes fully depend on the primary key. Additionally, it might already be in the third normal form (3NF) if there are no transitive dependencies, meaning no non-key attribute depends on another non-key attribute.

A screenshot of a computer

Description automatically generated

Figure 9.transaction table

Upon the successful completion of normalization, the normalized tables were thoroughly evaluated. It was found that they satisfactorily met the normalization standards as outlined by 1NF and 2NF; however, there was no need to extend the normalization process to the Third Normal Form (3NF). This achieved a significant reduction of redundancy and the enhancement of data integrity.

Now, the dataset transformed into a strong model that assists the bank in analytical processes and decision-making tasks.

1. **Non-Relational Data Modelling Techniques**

Scenario: Performance Analysis of Leading Japanese Automobile Brands

The given scenario focused on the performance analysis of leading Japanese automobile brands—Toyota, Mitsubishi, Honda, Mazda, and Nissan—the goal is to compile and analyse data that highlights how these brands compete in terms of pricing, performance, and fuel efficiency. This analysis is crucial for stakeholders looking to understand consumer preferences, and technological advancements among these leading brands. The use of a NoSQL database is suggested to handle the unstructured data associated with vehicle specifications and performance metrics.

The data analyst's role includes:

1. Aggregating vehicle specifications, including engine size, horsepower, and fuel efficiency, to compare performance across brands.

2. Analysing pricing data to understand how different models are positioned in the market and how pricing affects consumer choice.

3. Creating a robust, quarriable database that supports deep dives into specific areas of interest

I have initially created an ER Diagram in Visio to understand the structure of data and found the key entities and common attributes. ([ER Diagram of Leading Japanese Automobile Brands.vsdx](https://nzschoolofed-my.sharepoint.com/:u:/g/personal/764706847_nzse_ac_nz/EYS-8sZzkt5JjzvT7NvbyKIBFBsWLLkw_qxp-GLHNJPRCw?e=a1Hwm9))

Identified 5 key Entities and attributes:

1. Toyota
2. Mitsubishi
3. Honda
4. A computer screen shot of a computer

   Description automatically generatedMazda Nissan

Figure 10.Non-relational ER diagram

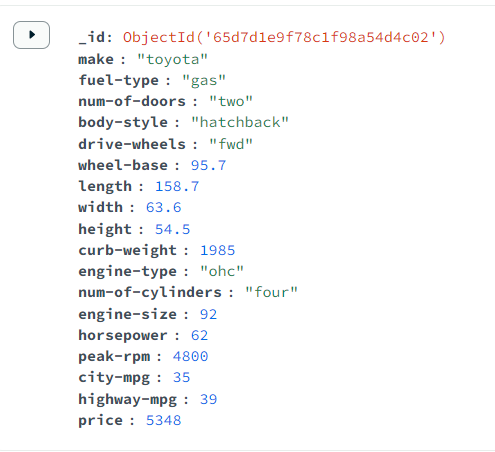
Non-Relational Data Modelling Techniques: I have created a Database in MongoDB Compass named as ‘GDDA707Assessment1’ and uploaded all 5 entities collections (Jason file) for my query and analysis.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer code

Description automatically generatedMongoDB database: data can be modelled to support flexible schemas which is ideal for compiling diverse data from multiple sources.

A screenshot of a computer code

Description automatically generatedBelow attached are the non-relational database collection’s structure (see Figure 10).

*Figure 11.collections*

1. **Documentation of Non-Relational Data Model Structure**

I have documented the non-relational data models structure clearly, encompassing entities,

attributes, and relationships. We have a total of 5 entities which are the topmost 5 automobile brands in Japan. Each attribute provides valuable information about the vehicle's design, capabilities, and efficiency.

Entities:

1. Toyota
2. Mitsubishi
3. Honda
4. Mazda
5. Nissan

Attributes and their relationship:

Make: The manufacturer of the vehicle. It helps identify the company that produced the car.

Fuel-Type: Indicates what kind of fuel the vehicle uses, such as gas, diesel or electric.

Num-of-Doors: The total number of doors on the vehicle

Body-Style: The general shape and style of the vehicle's body. Common styles include sedan, hatchback, wagon, and hardtop.

Drive-Wheels: The set of wheels that receive power from the engine. Common types are FWD (front-wheel drive), RWD (rear-wheel drive), and 4WD (four-wheel drive).

Wheelbase: The distance between the front and rear wheels of the vehicle.

Length, Width, Height: These dimensions define the overall size of the vehicle.

Curb-Weight: The weight of the vehicle without passengers or cargo, but with standard equipment and full fuel tank.

Engine-Type: Describes the configuration of the engine (ohc, dohc).

Num-of-Cylinders: The total number of cylinders in the engine. More cylinders generally provide more power but can reduce fuel efficiency.

Engine-Size: The total volume of the engine's cylinders, usually measured in cubic centimetres (cc) or litters.

Horsepower: A measure of the engine's power output. Higher horsepower indicates a more powerful engine capable of faster acceleration and higher top speeds.

Peak-RPM: The engine speed (revolutions per minute) at which the engine produces its maximum horsepower.

City-MPG and Highway-MPG: Miles per gallon ratings for city and highway driving, respectively.

Price: The retail cost of the vehicle. This can vary widely based on the make, model, features, and performance capabilities of the car.

These attributes collectively provide a comprehensive overview of a vehicle's performance, design, and operational costs, helping consumers and analysts compare different vehicles based on their specific needs and preferences.

**Task 2: Relational and Non-Relational Database Implementation**

1. **Implemented a relational database system.** 
   1. **Table Creation**

I have created tables in Google Cloud Big Query that corresponding to the identified and normalized entities for the fictional bank database “NZBank”. Entities/Tables are Bank, Customer, Account, Loan, and Transaction.

1. Open Big Query console by logging Google Cloud platform
2. Selected the GCP Project and Created a database and named as NZBank (student2023.NZBank)
3. Designed schema and created tables for each entity, by uploading the normalized data from python.

By following the above steps, I have successfully designed and created a NZBank database in Google Cloud Big Query where I can easily process my queries, data insertion and analysis. (see Figure)

A screenshot of a computer

Description automatically generated

Figure 12.Entity-Account

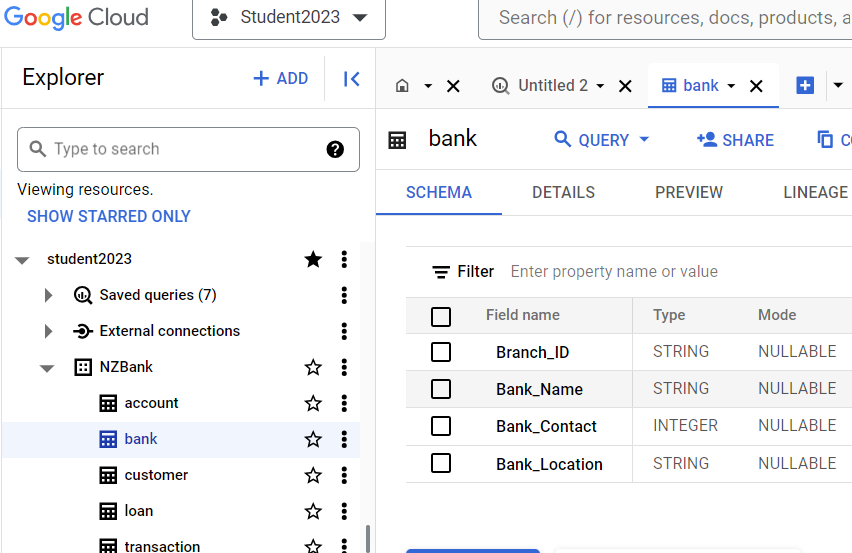


Figure 13.Entity-Bank

A screenshot of a computer

Description automatically generated

Figure 14.Entity-Customer

A screenshot of a computer

Description automatically generated

Figure 15.Entity-Loan

A screenshot of a computer

Description automatically generated

Figure 16.Entity-Transaction

* 1. **Record Insertion : Used INSERT INTO function**

1. Bank: I have successfully inserted 5 sample records to ‘bank’ table. (see Figure 17)

A screenshot of a computer

Description automatically generated

Figure 17.bank table

1. Customer: I have successfully inserted 5 sample records to ‘customer’ table. (see Figure 18)

A screenshot of a computer

Description automatically generated

Figure 18.cutomer table

1. Account: I have successfully inserted 5 sample records to ‘account’ table. (see Figure 19)

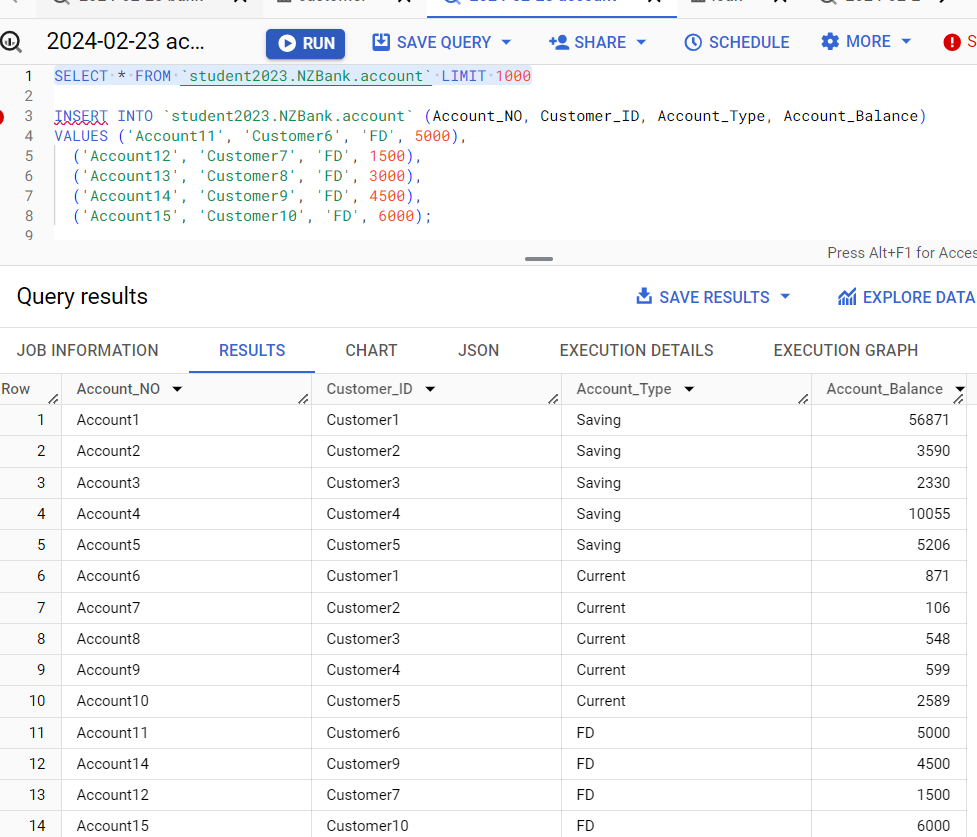


Figure 19.account table

1. Transaction: I have successfully inserted 5 sample records to ‘transaction’ table. (see Figure 20)

A screenshot of a computer

Description automatically generated

Figure 20.transaction table

1. Loan: I have successfully inserted 5 sample records to ‘loan’ table. (see Figure 21)

A screenshot of a computer

Description automatically generated

Figure 21.loan table

* 1. **Update Queries**

1. Customer table:

Mrs. Olivia Benson has requested an update on her current location.

Customer Name: Olivia Benson, Customer ID: Customer9, Current location: Dunedin

New location: Mangere

Please find below the queries for updating the customer table.

SELECT \* FROM student2023.NZBank.customer

1-- Updating 'customer' table

UPDATE student2023.NZBank.customer

SET Customer\_Name = 'Mangere'

WHERE Customer\_ID = 'Customer9'

--Above query was wrongly updated customer name instead customer location

2-- Updating 'customer name

UPDATE student2023.NZBank.customer

SET Customer\_Name = 'John Lawrance'

WHERE Customer\_ID = 'Customer9'

3--updating customer contact number

-- Updating 'customer contact number which has inserted wrongly.

UPDATE student2023.NZBank.customer

SET Customer\_Contact = 666666

WHERE Customer\_ID = 'Customer6'

4--updating customer contact number

UPDATE student2023.NZBank.customer

SET Customer\_Contact = 666667

WHERE Customer\_ID = 'Customer7'

5--updating customer contact number

UPDATE student2023.NZBank.customer

SET Customer\_Contact = 666668

WHERE Customer\_ID = 'Customer8'

6--updating customer contact number

UPDATE student2023.NZBank.customer

SET Customer\_Contact = 666669

WHERE Customer\_ID = 'Customer9'

7--updating customer contact number

UPDATE student2023.NZBank.customer

SET Customer\_Contact = 666670

WHERE Customer\_ID = 'Customer10'

A screenshot of a computer

Description automatically generated

1. Account table:

Mr. Alex Martin withdrawn $3500 from his FD account; bank requested an update on database.

Customer Name: Alex Martin, Customer ID: Customer6, FD withdrawal:3500

FD balance: 1500

1--updating FD deposit

UPDATE `student2023.NZBank.account`

SET Account\_Balance = 1500

WHERE Account\_NO = 'Account11'

Mr. Alex Martin withdrawn $3500 from his FD account; bank requested an update on database.

Customer Name: Alex Martin, Customer ID: Customer6, FD withdrawal:3500

FD balance: 1500

2--updating FD deposit

UPDATE `student2023.NZBank.account`

SET Account\_Balance = 500

WHERE Account\_NO = 'Account12'

Mr. Alex Martin withdrawn $3500 from his FD account; bank requested an update on database.

Customer Name: Alex Martin, Customer ID: Customer8, FD withdrawal:3500

FD balance: 1500

3--updating FD deposit

UPDATE `student2023.NZBank.account`

SET Account\_Balance = 5000

WHERE Account\_NO = 'Account13'

A screenshot of a computer

Description automatically generated

1. Loan table:

Mr. Johns Thomson settled his car loan; bank requested an update on database.

Customer Name: Johns Thomson, Customer ID: Customer5, Loan Type: Car Loan

Loan balance: 0

1--updating loan table

UPDATE `student2023.NZBank.loan`

SET Loan\_Amount = 0

WHERE Customer\_ID = 'Customer5'

--by mistake, above query updated both his car loan and education loan.

2--updating education loan balance.

UPDATE `student2023.NZBank.loan`

SET Loan\_Amount = 5500

WHERE Customer\_ID = 'Customer5' AND Loan\_Type = 'Education loan'

Mr. Miller Thompson settled his personal loan; bank requested an update on database.

Customer Name: Miller Thompson, Customer ID: Customer4, Loan Type: Personal Loan

Loan balance: 0

3--updating education loan balance.

UPDATE `student2023.NZBank.loan`

SET Loan\_Amount = 5500

WHERE Customer\_ID = 'Customer5' AND Loan\_Type = 'Education loan'

A screenshot of a computer

Description automatically generated

1. Transaction

Updating few transactions as per bank request.

1-- updating transaction table

  UPDATE `student2023.NZBank.transaction`

  SET Deposite = 500

  WHERE Transaction\_ID = 'T009'

2-- updating transaction table

   UPDATE `student2023.NZBank.transaction`

  SET Deposite = 3250

  WHERE Transaction\_ID = 'T005'

3-- updating transaction table

   UPDATE `student2023.NZBank.transaction`

  SET Deposite = 8050

  WHERE Transaction\_ID = 'T003'

A screenshot of a computer

Description automatically generated

1. Bank

Bank has instructed to change the location of major branches.

1--updating bank location

UPDATE `student2023.NZBank.bank`

SET Bank\_Location = 'Auckland'

WHERE Branch\_ID = 'NZB01'

2--updating bank location

UPDATE `student2023.NZBank.bank`

SET Bank\_Location = 'Wellington'

WHERE Branch\_ID = 'NZB03'

3--updating bank location

UPDATE `student2023.NZBank.bank`

SET Bank\_Location = 'Christ Church'

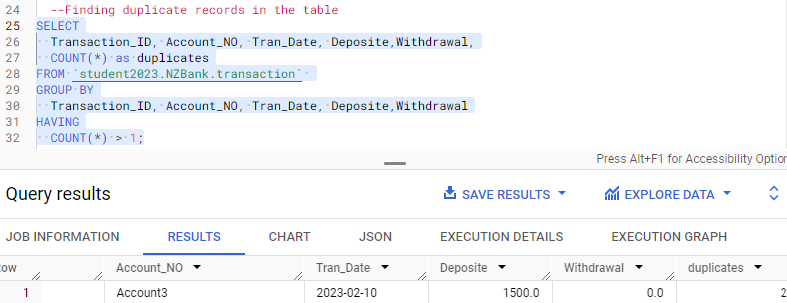
WHERE Branch\_ID = 'NZB05'

A screenshot of a computer

Description automatically generated

* 1. **Duplicate records removal**

1. In the transaction table, I found one duplicate record which was successfully removed using the query below.

Figure 22.transaction table

--delete all duplicate records

CREATE OR REPLACE TABLE `student2023.NZBank.transaction`  AS

SELECT \* EXCEPT(RowNumber)

FROM (

  SELECT \*,

    ROW\_NUMBER() OVER (PARTITION BY Transaction\_ID ORDER BY Transaction\_ID) AS RowNumber

  FROM  `student2023.NZBank.transaction`

)

WHERE RowNumber = 1;

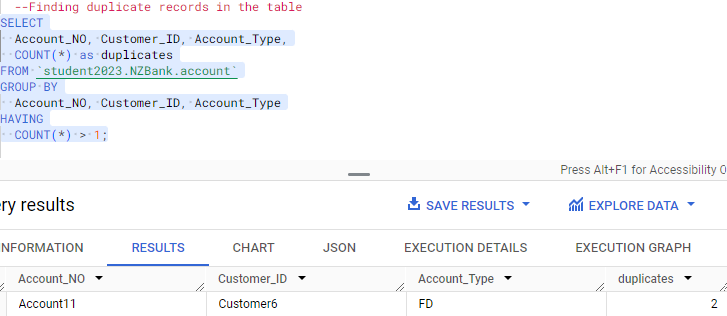
1. In the account table, I found one duplicate record which was successfully removed using the query below.

Figure 23.account table

--delete all duplicate records

CREATE OR REPLACE TABLE `student2023.NZBank.account`  AS

SELECT \* EXCEPT(RowNumber)

FROM (

  SELECT \*,

    ROW\_NUMBER() OVER (PARTITION BY Account\_NO ORDER BY Account\_NO) AS RowNumber

  FROM  `student2023.NZBank.account`

)

WHERE RowNumber = 1;

1. In the bank table, I found one duplicate record which was successfully removed using the query below.

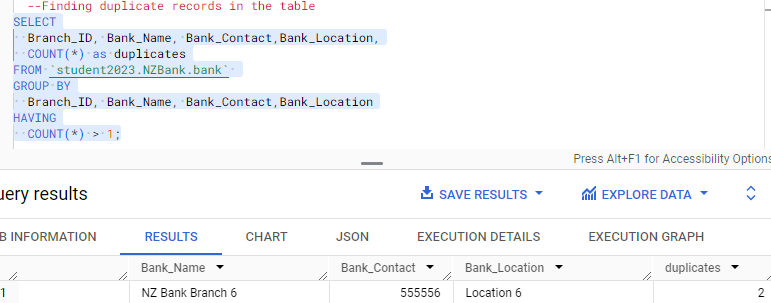


Figure 24.bank table

--delete all duplicate records

CREATE OR REPLACE TABLE `student2023.NZBank.bank`   AS

SELECT \* EXCEPT(RowNumber)

FROM (

  SELECT \*,

    ROW\_NUMBER() OVER (PARTITION BY Branch\_ID ORDER BY Branch\_ID) AS RowNumber

  FROM  `student2023.NZBank.bank`

)

WHERE RowNumber = 1;

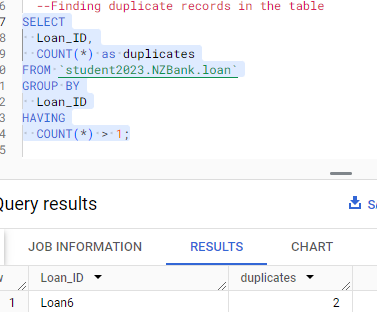
1. In the loan table, I found one duplicate record which was successfully removed using the query below.

Figure 25.loan table

--delete all duplicate records

CREATE OR REPLACE TABLE `student2023.NZBank.loan`   AS

SELECT \* EXCEPT(RowNumber)

FROM (

  SELECT \*,

    ROW\_NUMBER() OVER (PARTITION BY Loan\_ID ORDER BY Loan\_ID) AS RowNumber

  FROM  `student2023.NZBank.loan`

)

WHERE RowNumber = 1;

1. In the customer table, I found one duplicate record which was successfully removed using the query below.

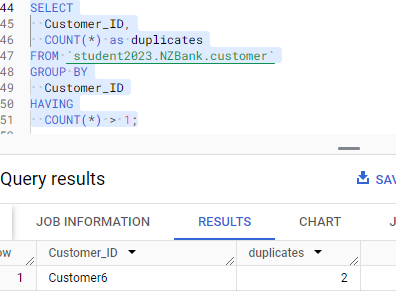


Figure 26.customer table

--delete all duplicate records

CREATE OR REPLACE TABLE `student2023.NZBank.customer`   AS

SELECT \* EXCEPT(RowNumber)

FROM (

  SELECT \*,

    ROW\_NUMBER() OVER (PARTITION BY Customer\_ID ORDER BY Customer\_ID) AS RowNumber

  FROM  `student2023.NZBank.customer`

)

WHERE RowNumber = 1;

* 1. **Table Joins for Data Aggregation**

1. Aggregate data across customers and their transactions. finding the total deposit and withdrawal amounts for each customer.

Our customer table has Customer\_ID as a primary key, and our transaction table includes Account\_NO and Customer\_ID (implicitly, through the account table which connects customers to their accounts).

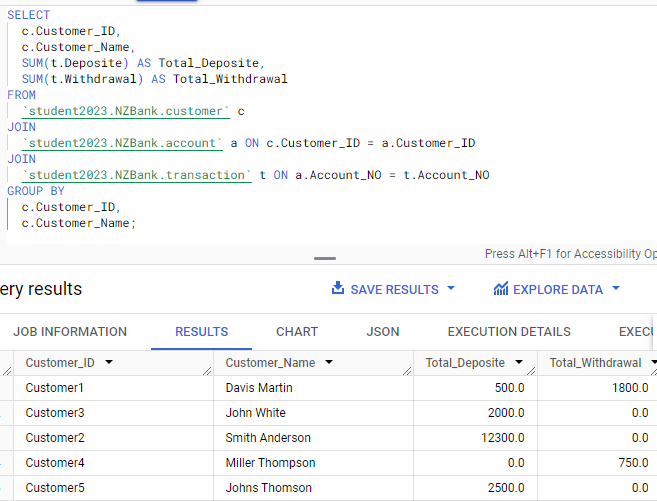
Result: Please see the table below.

Figure 27.Query1

Table 3.aggregation result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Row | Customer\_ID | Customer\_Name | Total\_Deposite | Total\_Withdrawal |
| 1 | Customer1 | Davis Martin | 500.0 | 1800.0 |
| 2 | Customer3 | John White | 2000.0 | 0.0 |
| 3 | Customer2 | Smith Anderson | 12300.0 | 0.0 |
| 4 | Customer4 | Miller Thompson | 0.0 | 750.0 |
| 5 | Customer5 | Johns Thomson | 2500.0 | 0.0 |

This query effectively aggregates deposit and withdrawal totals for each customer across all their accounts and transactions.

1. Collect the loan information, how many loans each customers taken, and the total loan amount borrowed.

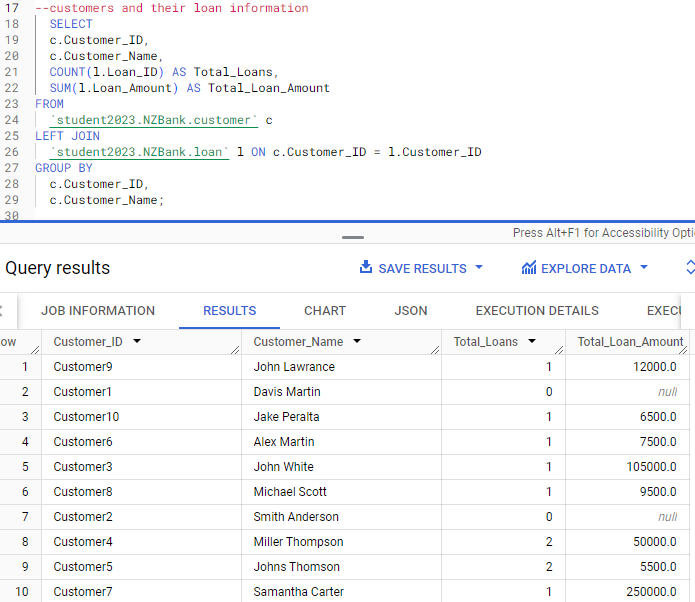


Figure 28.query2

Result: Please see the table below.

Table 4.result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Row | Customer\_ID | Customer\_Name | Total\_Loans | Total\_Loan\_Amount |
| 1 | Customer9 | John Lawrance | 1 | 12000.0 |
| 2 | Customer1 | Davis Martin | 0 | *null* |
| 3 | Customer10 | Jake Peralta | 1 | 6500.0 |
| 4 | Customer6 | Alex Martin | 1 | 7500.0 |
| 5 | Customer3 | John White | 1 | 105000.0 |
| 6 | Customer8 | Michael Scott | 1 | 9500.0 |
| 7 | Customer2 | Smith Anderson | 0 | *null* |
| 8 | Customer4 | Miller Thompson | 2 | 50000.0 |
| 9 | Customer5 | Johns Thomson | 2 | 5500.0 |
| 10 | Customer7 | Samantha Carter | 1 | 250000.0 |

This query is giving loan information by customer, number of loans each customer has taken out and the total amount borrowed. It's a useful query for analysing the lending aspect of the bank's operations.

1. Finding an overview of customer latest account balances. This query will involve the customer, account, and transaction tables.

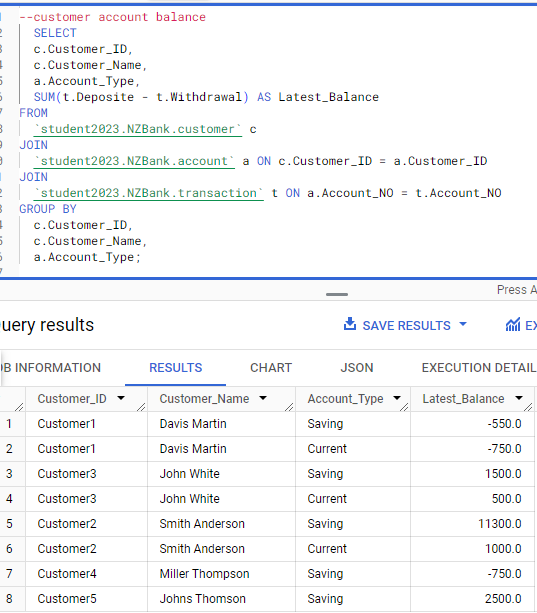


Figure 29.query 3

Result: Please see the table below.

Table 5.result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Row | Customer\_ID | Customer\_Name | Account\_Type | Latest\_Balance |
| 1 | Customer1 | Davis Martin | Saving | -550.0 |
| 2 | Customer1 | Davis Martin | Current | -750.0 |
| 3 | Customer3 | John White | Saving | 1500.0 |
| 4 | Customer3 | John White | Current | 500.0 |
| 5 | Customer2 | Smith Anderson | Saving | 11300.0 |
| 6 | Customer2 | Smith Anderson | Current | 1000.0 |
| 7 | Customer4 | Miller Thompson | Saving | -750.0 |
| 8 | Customer5 | Johns Thomson | Saving | 2500.0 |

This query provides an overview of each customer's account balances by account type, aggregating all transactions to calculate the latest balance.

# Implemented a non-relational database model.

1. **Collection Creation**

I have successfully transferred five collections of non-relational data into MongoDB Cloud, corresponding to the entities identified in the previous task. These collections, named Honda, Toyota, Nissan, Mitsubishi, and Mazda, offers documents detailing various characteristics of automobiles. Attributes covered include the make, fuel type, number of doors, body style, dimensions (such as wheelbase, length, width, and height), engine specifications (including type, number of cylinders, engine size, and horsepower), performance metrics (city and highway miles per gallon), and the price.

For a successful querying and analysis, these collections have been uploaded in a database named GDDA707Assessment1. This makes efficient data management and analysis across the different car makes represented in the collections.

The database is accessible through the MongoDB Atlas cluster at the following connection string: mongodb+srv://ajupeter:Student%402023@cluster0.kdgni2u.mongodb.net/.

This setup ensures that all data related to the project is centralized and readily available for further academic and analytical purposes.

A screenshot of a computer program

Description automatically generated

Figure 30.honda record

Figure 31.mazda record

A screenshot of a computer program

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generated

Figure 32.toyota record

Figure 33.nissan record

Figure 34.mitsubishi record

1. **Inserted a minimum of 5 sample documents for each entity collection by using “insertMany” function in “MONGOSH”.**
2. Inserted 5 sample record to Nissan collection. (Figure 36)
3. Inserted 5 sample record to Honda collection. (Figure 35)

A screen shot of a computer code

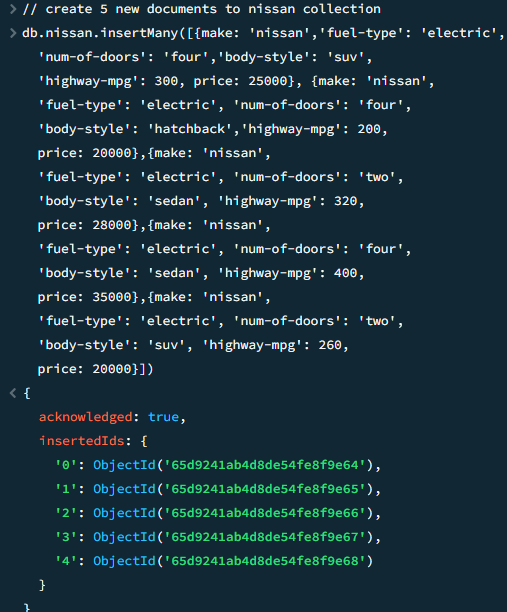
Description automatically generated

Figure 35.honda

Figure 36.nissan

1. Inserted 5 sample record to Mazda collection. (Figure 39)
2. Inserted 5 sample record to Toyota collection. (Figure 38)
3. Inserted 5 sample record to Mitsubishi collection. (Figure 37)

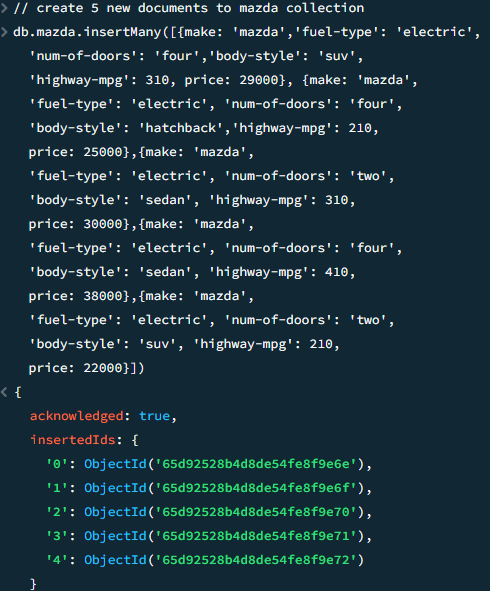
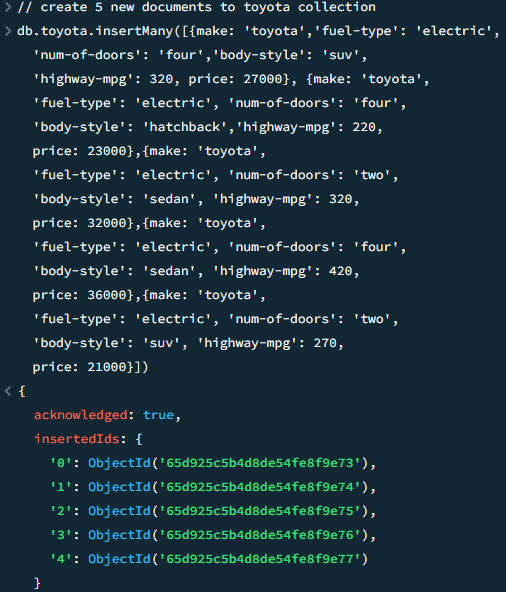
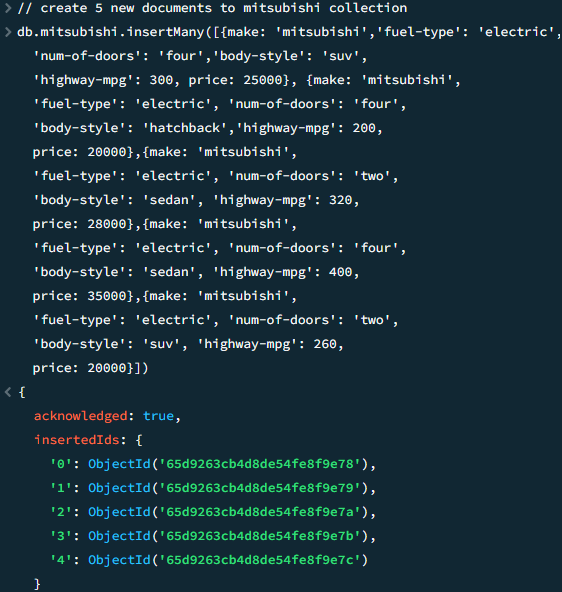


Figure 37.mitsubishi

Figure 38.toyota

Figure 39.mazda

1. **Update Queries by using “updateMany” function**

I have updated below 6 records in MongoDB database by using “MONGOSH”, please refer the images.

1. In Honda collection, the fuel type updated to ‘petrol’ from ‘gas’. (see Figure 41)
2. In Toyota collection, ‘fwd’ updated to ‘all wheel’ drive. (see Figure 40)
3. In Honda collection, fuel type changed to plug-in hybrid for vehicle id\_xxx (see Figure 38)
4. In Mazda collection, the fuel type updated to ‘petrol’ from ‘gas’. (see Figure 42)
5. In Nissan collection, ‘fwd’ updated to ‘all wheel’ drive. (see Figure 40)
6. In Mitsubishi collection, ‘fwd’ updated to ‘all wheel’ drive. (see Figure 41)

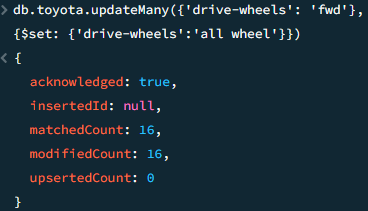
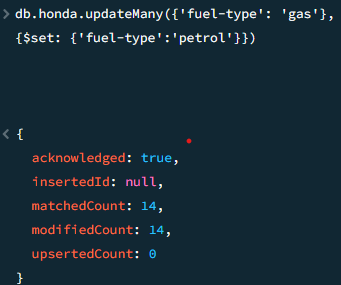
****

Figure 40.toyota drive-wheels updating

Figure 41.honda fuel-type updating

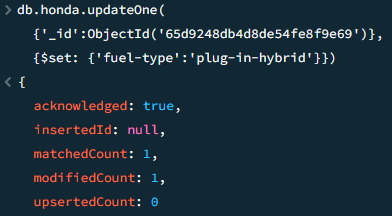
****

Figure 38. honda fuel-type updation

**A computer screen shot of a code

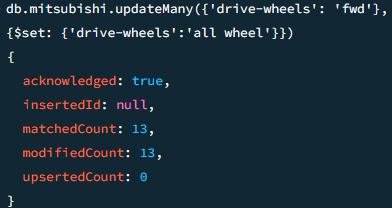
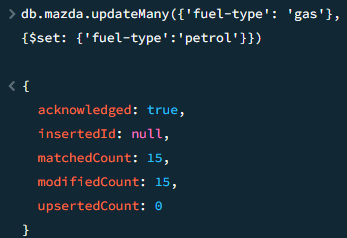
Description automatically generated******

Figure 41.mitsubishi drive-wheel updating

Figure 40. nissan drive-wheel updating

Figure 42.mazda fuel-type updating

1. **Duplicate records removal**
2. Nissan: I have successfully removed the duplicate documents from ‘Nissan’ collections.

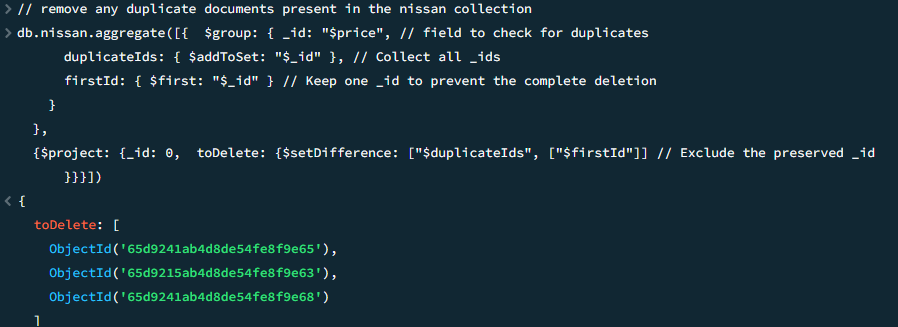


Figure 42

1. Honda: I have successfully removed the duplicate documents from ‘Honda’ collections.

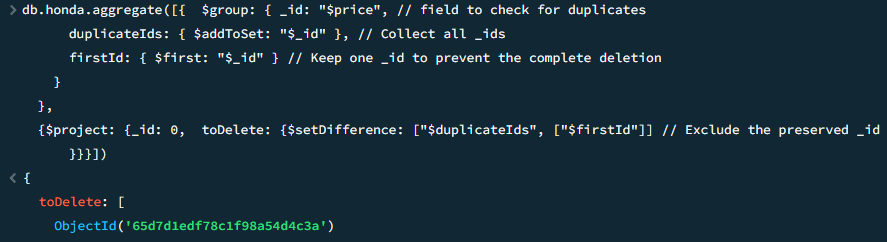


Figure 43

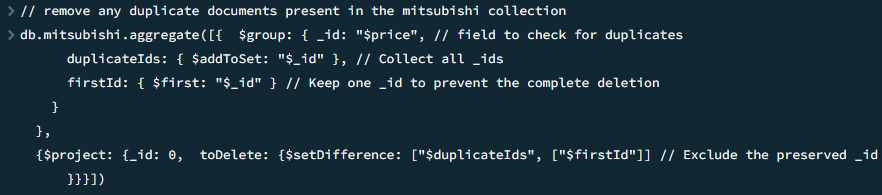
1. Toyota

No duplicate records were found.

1. Mazda

No duplicate records were found.

1. Mitsubishi: I have successfully removed the duplicate documents from ‘Mitsubishi’ collections.



A computer code with green text

Description automatically generated

1. **Data Retrieval and Aggregation**
2. Find vehicles with a 'fuel-type' of 'electric' in the 'Toyota', 'Mazda', and 'Nissan' collections?

I have attached a screenshot of the query and the first documents, which show that these mentioned brands offer a wide range of electric cars.

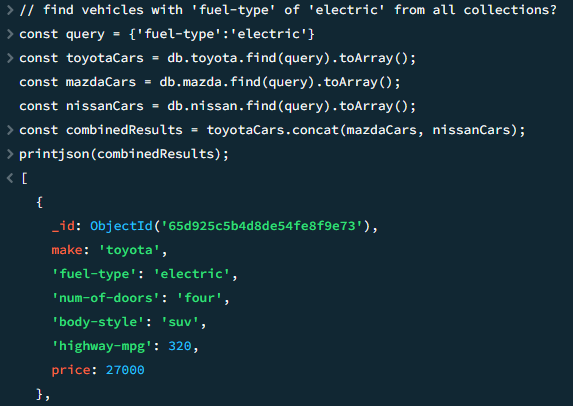


Figure 44.query result

1. Find cars that have a city mpg greater than 40 across all collections?

I have attached a screenshot of the query and the first documents, which show that these mentioned brands offer a wide range of electric cars.

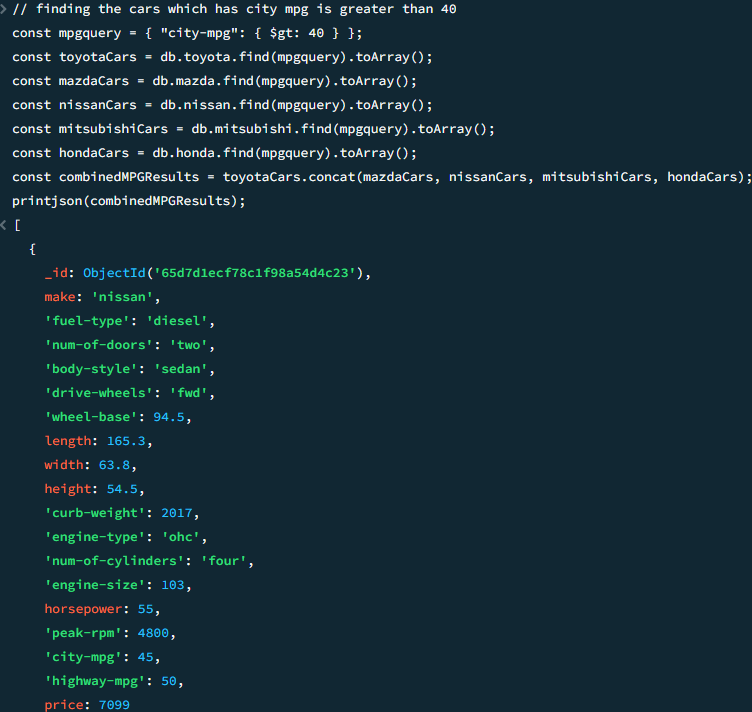


Figure 45.query result

1. Find the cheapest car options from 'Toyota', 'Mazda', and 'Nissan'? I have created a query to identify cars priced at less than $5,500.

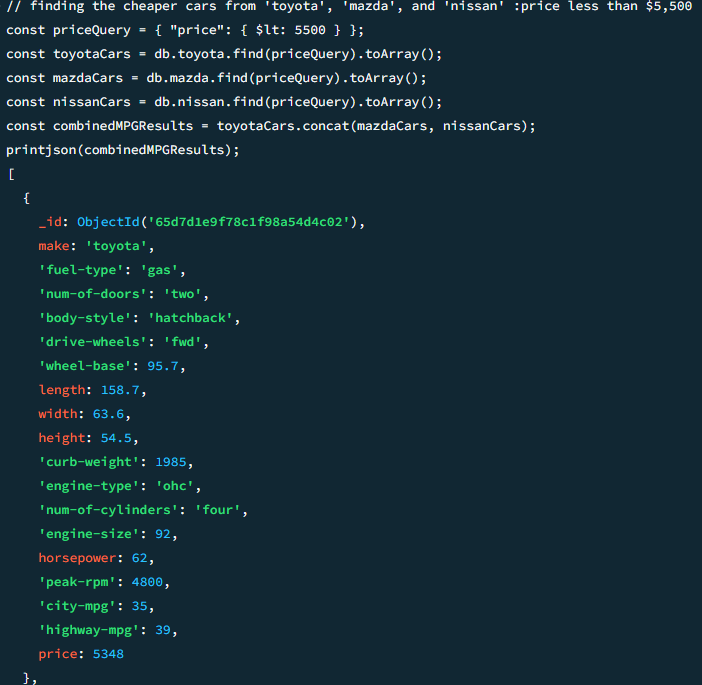


Figure 46.query result

**Documentation**

This documentation outlines the process undertaken to develop two distinct database solutions for separate scenarios: one involving a relational database for NZBank, a fictional New Zealand bank aiming to modernize its banking operations, and another involving a non-relational (NoSQL) database focusing on the performance analysis of leading Japanese automobile brands.

**Task 1: Relational Database for NZBank**

The bank recognized the need for a modern database solution to remain competitive and improve efficiency, security, and customer service.

Step-1: Process and Design Decisions

An ER model was developed using Visio, identifying five key entities and their relationships. The entities included Bank Branches, Customers, Accounts, Transactions, and Loans. The ER diagram (Figure 1) served as the foundational blueprint for the relational database design. It visually represented entities, attributes, and relationships, enable a clear understanding of the database structure.

Step-2: Rationale Behind Design Decisions

a) Normalization: The database was normalized to reduce redundancy and improve data integrity, ensuring efficient and compliant banking operations.

b) Entity Identification: Entities were chosen based on the core operations of banking, including customer management, account transactions, and loan processing.

Challenges and Solutions:

Challenge: Ensuring data integrity and relationship consistency among entities.

Solution: Implementing foreign key and carefully designing the relational schema to maintain data integrity.

**Task 2: Non-Relational Database for Automobile Performance Analysis**

The project aimed to analyse data on leading Japanese automobile brands (Toyota, Mitsubishi, Honda, Mazda, and Nissan) to assess pricing, performance, and fuel efficiency.

Step-1: Process and Design Decisions:

MongoDB database was selected to process the unstructured data associated with vehicle specifications and performance records.

Step-2: Data Modelling Techniques:

Data for each automobile brand was stored in separate collections, allowing for flexible schema and efficient storage of vehicle specifications.

Schema Design: While the collections Toyota, Mitsubishi, Honda, Mazda, and Nissan had similar attributes.

Challenges and Solutions:

Challenge: As a newcomer to MongoDB, I encountered difficulties in performing queries and analyses across multiple collections due to my lack of familiarity with the database.

Solution: Through continuous learning and practical experience, I gradually acquired the skills necessary to effectively query and analyse data within MongoDB.