Introduction-Case Study

LibraryManageSystem is the database for the college Sagarmatha College of Science and Technology in Sanepa,Kathmandu.This system allows to add books and journals in the library.This also allows adding Library members.Member can issue and return book and journals.If member did not return the book in deadline then fine amount will be added in the fine_amount table.

Problem Statement:

This LibraryManageSystem is a new system for this college to add the books,journals and members.They need this system to run daily library transaction tasks.

Requirements

This college requires a solution that would enable them to add books and journals in their database.As well as Add new members into the system.The solution must have following requirements:

i. Book: The solution must allow adding books in the system.

ii. Journals:The solution must allow adding Journals in the system.

iii. LibraryMember:The solution must allow adding library members in the. database.

iv. Issued: The solution must allow to Issue the book with member details

v . Return: The Solution must allow to Return the book

vi. FineAmount: The solution must allow to add if member did not return book or journal on deadline.

Solution

To meet the requirements of Sagarmatha College,we propose a system with the following features

1 .Books Table: The system will allow to add books in the system.

2. Journals Table :The system will allow to add journals in the system.

3. Member Table : The system will allow to add library members.

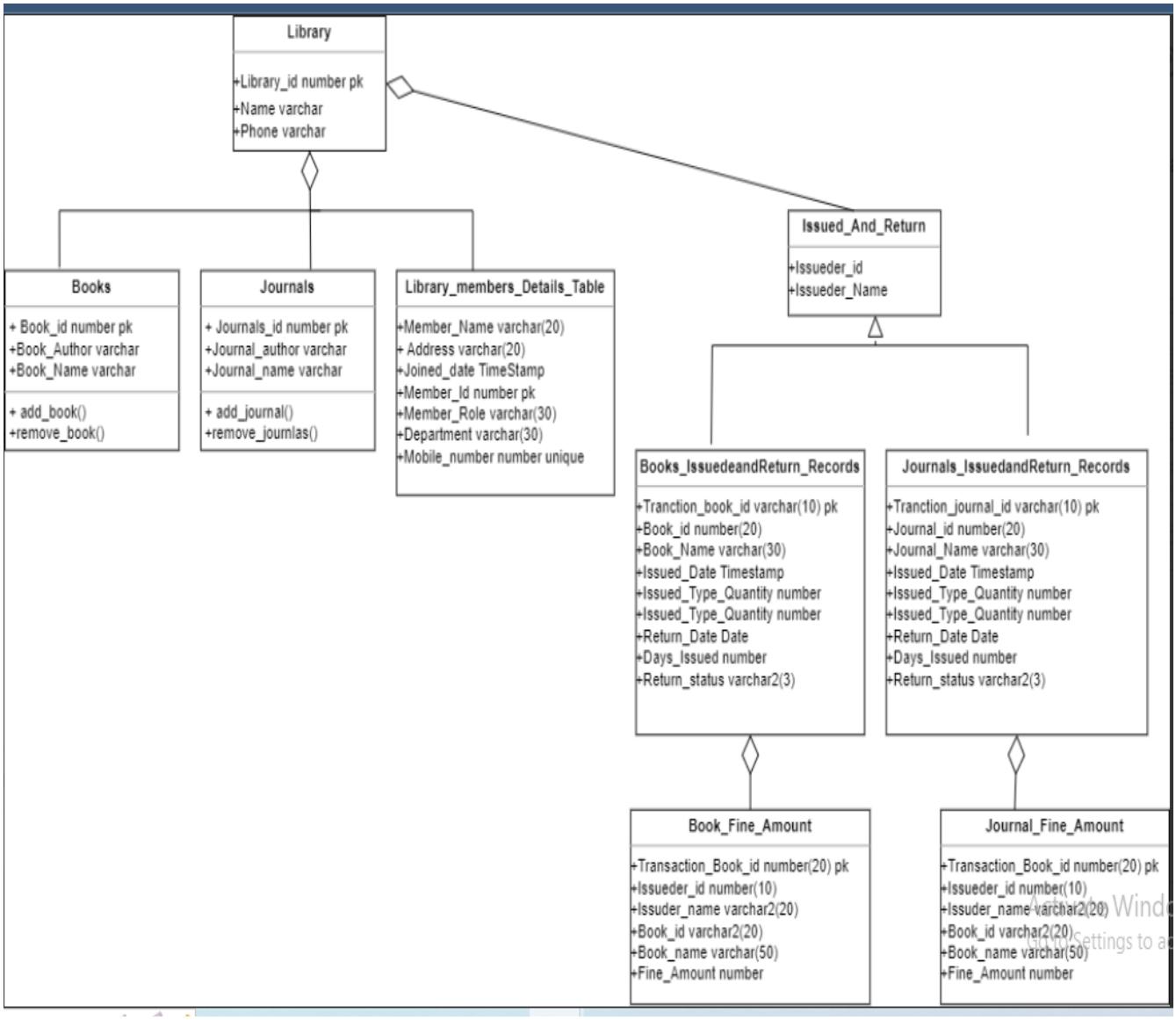
4 Issued Table : The system will allow to record book details and issueder details

5 Return Table : The system will record the book and member information who return .

6. Fine Amount : The system will add fine to the reader account if individual did not books and journals in time.

Task One : UML class Diagram

UML class Diagram of LibraryManagentSystem



Task Two:

Demonstration of SQL Code and its working.

a. Drop Tables, Procedures , Sequence and Triggers:

Query:

```
DROP TABLE Library;
DROP SEQUENCE boo_seq;

DROP TABLE Books;

DROP PROCEDURE add_book;
DROP PROCEDURE remove_book;


DROP TABLE Journals;
DROP PROCEDURE add_journal;
DROP PROCEDURE remove_journal;
```

Output:

```
Table dropped.

Sequence dropped.

Table dropped.

Procedure dropped.

Procedure dropped.

Table dropped.

Procedure dropped.

Procedure dropped.
```

Query:

```
14
15  DROP TABLE Library_Members_Details_Tables;
16  DROP TYPE Library_Members_Details;
17  DROP TYPE Library_Members;
```

Output:

```
Table dropped.

Type dropped.

Type dropped.
```

Query:

```
18
19  DROP TRIGGER update_return_status_book;
20  DROP TABLE BookS_FINEAMOUNT;
21  Drop TABLE BOOKS_ISSUEDANDRETURN_RECORDS;
22  DROP TYPE BOOKS_ISSUED_AND_RETURN_DETAILS;
23  DROP TYPE ISSUED_AND_RETURN;
24
25
```

Output:

```
Table dropped.

Table dropped.

Type dropped.

Type dropped.
```

Query:

```
26
27  DROP TRIGGER update_return_status_journals;
28  DROP TABLE JOURNALS_FINEAMOUNT;
29  DROP TABLE JOURNALS_ISSUEDANDRETURN_RECORDS;
30  DROP TYPE JOURNALS_ISSUED_AND_RETURN_DETAILS;
31  DROP TYPE ISSUED_AND_RETURN;
```

Output:

```
Trigger dropped.

Table dropped.

Table dropped.

Type dropped.

Type dropped.
```

b. Creating Tables,Sequence,procedure and types:

Ouery:

```
CREATE TABLE Library (
    Library_ID NUMBER(5) PRIMARY KEY,
    Name VARCHAR(50),
    Phone INTEGER
);

CREATE SEQUENCE boo_seq START WITH 1 INCREMENT BY 1;

CREATE TABLE Books (
    Book_ID NUMBER(5) PRIMARY KEY,
    Book_Author VARCHAR(25),
    Book_Name VARCHAR(25),
    Library_ID NUMBER(5),
    CONSTRAINT fk_library FOREIGN KEY (Library_ID) REFERENCES Library(Library_ID)
);
```

```
-- Create procedure with the correct sequence name
CREATE OR REPLACE PROCEDURE add_book(
    p_Book_ID NUMBER,
    p_Library_ID NUMBER,
    p_Book_Author VARCHAR,
    p_Book_Name VARCHAR
) AS
BEGIN
    INSERT INTO Books (Book_ID, Book_Author, Book_Name, Library_ID)
    VALUES (p_Book_ID, p_Book_Author, p_Book_Name, p_Library_ID);
END add_book;
```

Output:

```
Table created.

Sequence created.

Table created.

Procedure created.
```

Query:

```
CREATE TABLE Journals (
    Journal_ID NUMBER(5) PRIMARY KEY,
    Journal_Author VARCHAR(100),
    Journal_Name VARCHAR(100),
    Library_ID NUMBER(5)
    -- CONSTRAINT fk_j FOREIGN KEY (Library_ID) REFERENCES Library(Library_ID)
);
CREATE OR REPLACE PROCEDURE remove_journal(
    p_Journal_ID NUMBER
) AS
BEGIN
    DELETE FROM Journals  WHERE Journal_ID = p_Journal_ID;
    COMMIT; -- Ensure the changes are committed
    DBMS_OUTPUT.PUT_LINE('Journal  removed: ' || p_Journal_ID);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Journal not found with Journal_ID: ' || p_Journal_ID);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error removing Journal: ' || SQLERRM);
END remove_journal;
/
```

```
CREATE OR REPLACE PROCEDURE add_journal(
    p_Journal_ID NUMBER,
    p_Library_ID NUMBER,
    p_Journal_Author VARCHAR,
    p_Journal_Name VARCHAR
) AS
BEGIN
    INSERT INTO Journals (Journal_ID, Journal_Author, Journal_Name, Library_ID)
    VALUES (p_Journal_ID, p_Journal_Author, p_Journal_Name, p_Library_ID);
END add_journal;
```

Output:

```
Table created.

Procedure created.

Procedure created.
```

Query:

```
create or replace type Library_Members as object(
        Member_Name varchar(30),
        Address varchar(30),
        Joined_date TimeStamp
)Not final;

CREATE or REPLACE TYPE Library_Members_Details  under Library_Members(
    Member_ID number(20),
    Member_Role Varchar(30),
    Department Varchar(30),
    Mobile_number number(15)

);
Create Table Library_Members_Details_Tables of Library_Members_Details(primary key(Member_ID),unique(Mobile_number));
```

Outputs:

```
Type created.
```

```
Type created.
```

```
Table created.
```

Query:

```
248 create or replace type ISSUED_AND_RETURN as object(
249       ISSUEDER_ID NUMBER(10) ,
250       ISSUEDER_NAME VARCHAR2(20)
251   )Not final;
252
253 CREATE or REPLACE TYPE BOOKS_ISSUED_AND_RETURN_DETAILS  under ISSUED_AND_RETURN(
254       TRANCTION_BOOK_ID VARCHAR(10),
255       BOOK_ID NUMBER(20),
256       BOOK_NAME VARCHAR(30),
257       ISSUED_DATE TIMESTAMP,
258       ISSUED_TYPE_QUANTITY NUMBER(10),
259       RETRUN_DATE DATE,
260       DAYS_ISSUED NUMBER,
261       RETURN_STATUS VARCHAR2(3)
262   );
```

```
263 Create Table BOOKS_ISSUEDANDRETURN_RECORDS of BOOKS_ISSUED_AND_RETURN_DETAILS(primary key(TRANCTION_BOOK_ID));
264
265 CREATE TABLE BOOKS_FINEAMOUNT (
266     TRANCTION_BOOK_ID number(20) primary key  ,
267     ISSUEDER_ID NUMBER(10)  ,
268     ISSUEDER_NAME VARCHAR2(20),
269     BOOK_ID VARCHAR2(20),
270     BOOK_NAME VARCHAR(50),
271     FINE_AMOUNT NUMBER
272 );
```

Output:

```
Type created.
```

```
Type created.
```

```
Table created.

Table created.
```

Query:

```
CREATE or REPLACE TYPE JOURNALS_ISSUED_AND_RETURN_DETAILS  under ISSUED_AND_RETURN(
    TRANCTION_JOURNAL_ID VARCHAR(10),
    JOURNAL_ID NUMBER(20),
    JOURNAL_NAME VARCHAR(30),
    ISSUED_DATE TIMESTAMP,
    ISSUED_TYPE_QUANTITY NUMBER(10),
    RETRUN_DATE DATE,
    DAYS_ISSUED NUMBER,
    RETURN_STATUS VARCHAR2(3)
);
```

```
CREATE TABLE JOURNALS_FINEAMOUNT (
    TRANCTION_JOURNAL_ID VARCHAR(10) primary key,
    ISSUEDER_ID NUMBER(10)  ,
    ISSUEDER_NAME VARCHAR2(20),
    JOURNAL_ID VARCHAR2(20),
    JOURNAL_NAME VARCHAR(50),
    FINE_AMOUNT NUMBER
);
Create Table JOURNALS_ISSUEDANDRETURN_RECORDS of JOURNALS_ISSUED_AND_RETURN_DETAILS( primary key(TRANCTION_JOURNAL_ID));
```

Output:

```
Type created.
```

```
Table created.

Table created.
```

c. Creating Triggers:

Query:

```
CREATE OR REPLACE TRIGGER update_return_status_book
BEFORE INSERT OR UPDATE ON BOOKS_ISSUEDANDRETURN_RECORDS
FOR EACH ROW
DECLARE
    v_days_diff NUMBER;
BEGIN
    IF :NEW.RETRUN_DATE IS NOT NULL THEN
        -- Calculate days as a number
        v_days_diff := EXTRACT(DAY FROM (:NEW.RETRUN_DATE - :NEW.ISSUED_DATE));
        :NEW.DAYS_ISSUED := v_days_diff;
        :NEW.RETURN_STATUS := 'YES';
    ELSE
        -- Calculate days as a number
        v_days_diff := EXTRACT(DAY FROM (SYSDATE - :NEW.ISSUED_DATE));
        :NEW.DAYS_ISSUED := v_days_diff;
        :NEW.RETURN_STATUS := 'NO';
    END IF;

    -- Check if fine is needed based on the range of days issued
    IF v_days_diff > 10 AND v_days_diff <= 20 THEN
        INSERT INTO BOOKS_FINEAMOUNT (TRANCTION_BOOK_ID,ISSUEDER_ID, ISSUEDER_NAME, BOOK_ID, BOOK_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_BOOK_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.BOOK_ID, :NEW.BOOK_NAME, 50);
    ELSIF v_days_diff > 20 AND v_days_diff <= 30 THEN
        INSERT INTO BOOKS_FINEAMOUNT (TRANCTION_BOOK_ID,ISSUEDER_ID, ISSUEDER_NAME, BOOK_ID, BOOK_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_BOOK_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.BOOK_ID, :NEW.BOOK_NAME, 100);
    ELSE
        INSERT INTO BOOKS_FINEAMOUNT (TRANCTION_BOOK_ID,ISSUEDER_ID, ISSUEDER_NAME, BOOK_ID, BOOK_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_BOOK_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.BOOK_ID, :NEW.BOOK_NAME, 500);
    END IF;
END update_return_status_book;
/
```

Output:

Outputs:

Query:

```
CREATE OR REPLACE TRIGGER update_return_status_journals
BEFORE INSERT OR UPDATE ON JOURNALS_ISSUEDANDRETURN_RECORDS
FOR EACH ROW
DECLARE
    v_days_diff NUMBER;
BEGIN
    IF :NEW.RETRUN_DATE IS NOT NULL THEN
        -- Calculate days as a number
        v_days_diff := EXTRACT(DAY FROM (:NEW.RETRUN_DATE - :NEW.ISSUED_DATE));
        :NEW.DAYS_ISSUED := v_days_diff;
        :NEW.RETURN_STATUS := 'YES';
    ELSE
        -- Calculate days as a number
        v_days_diff := EXTRACT(DAY FROM (SYSDATE - :NEW.ISSUED_DATE));
        :NEW.DAYS_ISSUED := v_days_diff;
        :NEW.RETURN_STATUS := 'NO';
    END IF;
```

```
    -- Check if fine is needed based on the range of days issued
    IF v_days_diff > 10 AND v_days_diff < 20 THEN
        INSERT INTO JOURNALS_FINEAMOUNT (TRANCTION_JOURNAL_ID,ISSUEDER_ID, ISSUEDER_NAME, JOURNAL_ID, JOURNAL_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_JOURNAL_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.JOURNAL_ID, :NEW.JOURNAL_NAME, 100);
    ELSIF v_days_diff >= 20 AND v_days_diff < 30 THEN
        INSERT INTO JOURNALS_FINEAMOUNT (TRANCTION_JOURNAL_ID,ISSUEDER_ID, ISSUEDER_NAME, JOURNAL_ID, JOURNAL_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_JOURNAL_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.JOURNAL_ID, :NEW.JOURNAL_NAME, 200);
    ELSE
        INSERT INTO JOURNALS_FINEAMOUNT (TRANCTION_JOURNAL_ID,ISSUEDER_ID, ISSUEDER_NAME, JOURNAL_ID, JOURNAL_NAME, FINE_AMOUNT)
        VALUES (:NEW.TRANCTION_JOURNAL_ID,:NEW.ISSUEDER_ID, :NEW.ISSUEDER_NAME, :NEW.JOURNAL_ID, :NEW.JOURNAL_NAME, 800);
    END IF;
END update_return_status_journals;
/
```

Output:

d. Now we will insert values in all Tables:

Query:

```
DECLARE
    v_Library_ID NUMBER := 05; -- Use the appropriate Library_ID
    v_Book_Details SYS.ODCIVARCHAR2LIST := SYS.ODCIVARCHAR2LIST(
        'Ram Kumar|DSA',
        'Shyam Hari|DataBase',
        'Alice Poudel|C++',
        'Pritam Kumar|Linear Algebra',
        'Purna Bdr Kc|Statistics'
        -- Add more book details as needed
    );
    v_Book_Author VARCHAR2(25);
    v_Book_Name VARCHAR2(25);
    v_Book_ID NUMBER;
BEGIN
    -- Loop through the book details and add each book
    FOR i IN 1..v_Book_Details.COUNT LOOP
        -- Split the book details into author and title
        SELECT REGEXP_SUBSTR(v_Book_Details(i), '[^|]+', 1, 1) INTO v_Book_Author FROM DUAL;
        SELECT REGEXP_SUBSTR(v_Book_Details(i), '[^|]+', 1, 2) INTO v_Book_Name FROM DUAL;

        -- Specify the Book_ID (replace with your desired value)
        v_Book_ID := 100 + i; -- Replace with your desired logic for Book_ID

        -- Call the add_book procedure with the specified Book_ID
        add_book(v_Book_ID, v_Library_ID, v_Book_Author, v_Book_Name);
```

```
97  select * from Books;
98
```

Output:

BOOK_ID	BOOK_AUTHOR	BOOK_NAME	LIBRARY_ID
101	Ram Kumar	DSA	5
102	Shyam Hari	DataBase	5
103	Alice Poudel	C++	5
104	Pritam Kumar	Linear Algebra	5
105	Purna Bdr Kc	Statistics	5

Query:

```
DECLARE
    v_Library_ID NUMBER := 05; -- Use the appropriate Library_ID
    v_Journal_Details SYS.ODCIVARCHAR2LIST := SYS.ODCIVARCHAR2LIST(
        'Pratiba Patel|Web Developemnt',
        'Alice Pratap|Face Recognition',
        'Sushma Kiran|AI',
        'Dina Nath|ChatBot'
    );
    v_Journal_Author VARCHAR2(100);
    v_Journal_Name VARCHAR2(100);
    v_Journal_ID NUMBER;
BEGIN
    -- Loop through the journal details and add each journal
    FOR i IN 1..v_Journal_Details.COUNT LOOP
        -- Split the journal details into author and title
        SELECT REGEXP_SUBSTR(v_Journal_Details(i), '[^|]+', 1, 1) INTO v_Journal_Author FROM DUAL;
        SELECT REGEXP_SUBSTR(v_Journal_Details(i), '[^|]+', 1, 2) INTO v_Journal_Name FROM DUAL;

        -- Specify the Journal_ID (replace with your desired value)
        v_Journal_ID := 200 + i; -- Replace with your desired logic for Journal_ID

        -- Call the add_journal procedure with the specified Journal_ID
        add_journal(v_Journal_ID, v_Library_ID, v_Journal_Author, v_Journal_Name);
    END LOOP;
END;
```

```
SELECT * FROM JOURNALS;
```

Output:

JOURNAL_ID	JOURNAL_AUTHOR	JOURNAL_NAME	LIBRARY_ID
201	Pratiba Patel	Web Developemnt	5
202	Alice Pratap	Face Recognition	5
203	Sushma Kiran	AI	5
204	Dina Nath	ChatBot	5

Query:

```
224 INSERT INTO Library_Members_Details_Tables VALUES ('Ram Pratap','Kathmandu',SYSTIMESTAMP,001,'Teacher','IT',9843215987);
225 INSERT INTO Library_Members_Details_Tables VALUES ('Bikas Nath','Birgunj',SYSTIMESTAMP,002,'Student','IT',9843215887);
226 INSERT INTO Library_Members_Details_Tables VALUES ('Sita Nuepane','Sagarmatha',SYSTIMESTAMP,003,'Student','Engineering',9843217987);
227 INSERT INTO Library_Members_Details_Tables VALUES ('Prabesh Bhujel','Kathmandu',SYSTIMESTAMP,004,'Teacher','Engineering',9847215987);
228 INSERT INTO Library_Members_Details_Tables VALUES ('Shyam Kumar','Pokhara',SYSTIMESTAMP,005,'Teacher','IT',98437715987);
229
```

```
230 select * from Library_Members_Details_Tables;
231
```

Outputs:

MEMBER_NAME	ADDRESS	JOINED_DATE	MEMBER_ID	MEMBER_ROLE	DEPARTMENT	MOBILE_NUMBER
Ram Pratap	Kathmandu	26-DEC-23 07.14.14.830687 PM	1	Teacher	IT	9843215987
Bikas Nath	Birgunj	26-DEC-23 07.14.14.838913 PM	2	Student	IT	9843215887
Sita Nuepane	Sagarmatha	26-DEC-23 07.14.14.842792 PM	3	Student	Engineering	9843217987
Prabesh Bhujel	Kathmandu	26-DEC-23 07.14.14.846512 PM	4	Teacher	Engineering	9847215987
Shyam Kumar	Pokhara	26-DEC-23 07.14.14.850384 PM	5	Teacher	IT	98437715987

Query:

```
/
INSERT INTO BOOKS_ISSUEDANDRETURN_RECORDS(TRANCTION_BOOK_ID,ISSUEDER_ID,ISSUEDER_NAME,BOOK_ID,BOOK_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (1,001,'Ram Kumar',101,'DSA',TO_TIMESTAMP('2022-09-01 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-21', 'YYYY-MM-DD'));
INSERT INTO BOOKS_ISSUEDANDRETURN_RECORDS(TRANCTION_BOOK_ID,ISSUEDER_ID,ISSUEDER_NAME,BOOK_ID,BOOK_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (2,002,'Bikas Nath',103,'C++    ',TO_TIMESTAMP('2022-09-06 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-26', 'YYYY-MM-DD'));
INSERT INTO BOOKS_ISSUEDANDRETURN_RECORDS(TRANCTION_BOOK_ID,ISSUEDER_ID,ISSUEDER_NAME,BOOK_ID,BOOK_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (3,003,'Sita Nuepane    ',102,'DataBase',TO_TIMESTAMP('2022-09-07 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-8', 'YYYY-MM-DD'));
INSERT INTO BOOKS_ISSUEDANDRETURN_RECORDS(TRANCTION_BOOK_ID,ISSUEDER_ID,ISSUEDER_NAME,BOOK_ID,BOOK_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (4,004,'Prabesh Bhujel  ',105,'Statistics',TO_TIMESTAMP('2022-09-09 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2023-01-21', 'YYYY-MM-DD'));
```

```
SELECT * FROM BOOKS_ISSUEDANDRETURN_RECORDS;
SELECT * FROM BOOKS_FINEAMOUNT;
```

Output:

ISSUEDER_ID	ISSUEDER_NAME	TRANCTION_BOOK_ID	BOOK_ID	BOOK_NAME	ISSUED_DATE	ISSUED_TYPE_QUANTITY	RETRUN_DATE	DAYS_ISSUED	RETURN_STATUS
1	Ram Kumar	1	101	DSA	01-SEP-22 12.00.00.000000 PM	1	21-SEP-22	19	YES
2	Bikas Nath	2	103	C++	06-SEP-22 12.00.00.000000 PM	1	26-SEP-22	19	YES
3	Sita Nuepane	3	102	DataBase	07-SEP-22 12.00.00.000000 PM	1	08-SEP-22	0	YES
4	Prabesh Bhujel	4	105	Statistics	09-SEP-22 12.00.00.000000 PM	1	21-JAN-23	133	YES

Download CSV

4 rows selected.

TRANCTION_BOOK_ID	ISSUEDER_ID	ISSUEDER_NAME	BOOK_ID	BOOK_NAME	FINE_AMOUNT
1	1	Ram Kumar	101	DSA	50
2	2	Bikas Nath	103	C++	50
3	3	Sita Nuepane	102	DataBase	500
4	4	Prabesh Bhujel	105	Statistics	500

Query:

```
/
INSERT INTO JOURNALS_ISSUEDANDRETURN_RECORDS(TRANCTION_JOURNAL_ID,ISSUEDER_ID,ISSUEDER_NAME,JOURNAL_ID,JOURNAL_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (1,001,'Ram Pratap  ',201,'Web Developemnt',TO_TIMESTAMP('2022-09-01 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-21', 'YYYY-MM-DD'));
INSERT INTO JOURNALS_ISSUEDANDRETURN_RECORDS(TRANCTION_JOURNAL_ID,ISSUEDER_ID,ISSUEDER_NAME,JOURNAL_ID,JOURNAL_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (2,003,'Sita Nuepane',203,'AI',TO_TIMESTAMP('2022-09-03 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-5', 'YYYY-MM-DD'));
INSERT INTO JOURNALS_ISSUEDANDRETURN_RECORDS(TRANCTION_JOURNAL_ID,ISSUEDER_ID,ISSUEDER_NAME,JOURNAL_ID,JOURNAL_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (3,002,'Bikas Nath',204,'ChatBot',TO_TIMESTAMP('2022-09-10 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2022-09-30', 'YYYY-MM-DD'));
INSERT INTO JOURNALS_ISSUEDANDRETURN_RECORDS(TRANCTION_JOURNAL_ID,ISSUEDER_ID,ISSUEDER_NAME,JOURNAL_ID,JOURNAL_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (4,004,'Prabesh Bhujel  ',202,'Face Recognition ',TO_TIMESTAMP('2022-09-15 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2023-1-16', 'YYYY-MM-DD'));
INSERT INTO JOURNALS_ISSUEDANDRETURN_RECORDS(TRANCTION_JOURNAL_ID,ISSUEDER_ID,ISSUEDER_NAME,JOURNAL_ID,JOURNAL_NAME,ISSUED_DATE,ISSUED_TYPE_QUANTITY,RETRUN_DATE) VALUES
    (5,005,'Shyam Kumar',204,'ChatBot',TO_TIMESTAMP('2022-09-15 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),1,TO_DATE('2023-01-02', 'YYYY-MM-DD'));
select * from JOURNALS_ISSUEDANDRETURN_RECORDS;
select * from JOURNALS_FINEAMOUNT;

```

Output:

SQL Worksheet Clear Find Actions Save Run

ISSUEDER_ID	ISSUEDER_NAME	TRANCTION_JOURNAL_ID	JOURNAL_ID	JOURNAL_NAME	ISSUED_DATE	ISSUED_TYPE_QUANTITY	RETRUN_DATE	DAYS_ISSUED	RETURN_STATUS
1	Ram Pratap	1	201	Web Developemnt	01-SEP-22 12.00.00.000000 PM	1	21-SEP-22	19	YES
3	Sita Nuepane	2	203	AI	03-SEP-22 12.00.00.000000 PM	1	05-SEP-22	1	YES
2	Bikas Nath	3	204	ChatBot	10-SEP-22 12.00.00.000000 PM	1	30-SEP-22	19	YES
4	Prabesh Bhujel	4	202	Face Recognition	15-SEP-22 12.00.00.000000 PM	1	16-JAN-23	122	YES
5	Shyam Kumar	5	204	ChatBot	15-SEP-22 12.00.00.000000 PM	1	02-JAN-23	108	YES

TRANCTION_JOURNAL_ID	ISSUEDER_ID	ISSUEDER_NAME	JOURNAL_ID	JOURNAL_NAME	FINE_AMOUNT
1	1	Ram Pratap	201	Web Developemnt	100
2	3	Sita Nuepane	203	AI	800
3	2	Bikas Nath	204	ChatBot	100
4	4	Prabesh Bhujel	202	Face Recognition	800
5	5	Shyam Kumar	204	ChatBot	800

Download CSV

The script initializes by dropping existing tables, sequences, types, procedures, and triggers associated with the library management system. It then proceeds to create essential sequences for entities like Library, Books, Journals, Library Member Details, Book Issued and Return Details, Journal Issued and Return Details, Book Fine Amount, and Journal Fine Amount.

The Library table is defined with attributes such as Library_ID, Name, and Phone. It creates the Books table with a Book_ID sequence and associated procedures for adding and removing books. Similarly, the Journals table is created with relevant attributes.

The script also establishes the structure for library member details, including a hierarchical type with Member_ID, Member_Role, and Department. Further, it defines sequences for transactions related to book and journal issuance and return.

Triggers are implemented to automatically update return status and calculate fine amounts based on the duration of issuance.

Finally, foreign key constraints are added to ensure referential integrity, linking tables such as Books, Journals, Book Issued and Return Records, Journal Issued and Return Records, Book Fine Amount, and Journal Fine Amount with Library Member Details.

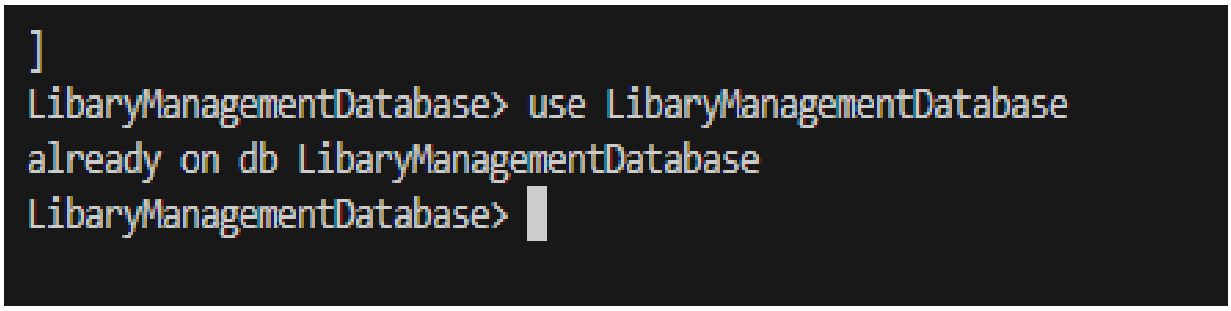
In summary, the script sets up a comprehensive database schema for a library management system, covering various entities and their relationships.

Task Three: Demonstration of MongoDB Scripts and its working

Using LibaryManagementDatabase

Query:
use LibaryManagementDatabase

Output:



Query:

```
db.Library.drop()
db.Books.drop()
db.Journals.drop()
db.LibraryMembersDetails.drop()
db.Books_Issuedandreturn_Records.drop()
db.Books_Fine_Amount.drop()
db.Journals_Issuedandreturn_Records.drop()
db.Journals_Fine_Amount.drop()
```

Output:

```
LibaryManagementDatabase> db.Library.drop()
true
LibaryManagementDatabase> db.Books.drop()
true
LibaryManagementDatabase> db.Journals.drop()
true
LibaryManagementDatabase> db.LibraryMembersDetails.drop()
true
LibaryManagementDatabase> db.Books_Issuedandreturn_Records.drop()
true
LibaryManagementDatabase> db.Books_Fine_Amount.drop()
true
LibaryManagementDatabase> db.Journals_Issuedandreturn_Records.drop()
true
LibaryManagementDatabase> db.Journals_Fine_Amount.drop()
true
```

Creating Collection Library:

```
db.createCollection("Library")
```

```
db.Library.insertOne({
    Library_ID: 5,
    Name: 'IT_Library',
    Phone: 557894
 });
```

Output:

```
db.Library.insertOne({
...     Library_ID: 5,
...     Name: 'IT_Library',
...   });
{
  acknowledged: true,
  insertedId: ObjectId("658b323fa09e7459e34ef9ad")
}
```

Creating Collection Books:

```
db.createCollection("Books")

const booksData = [
   { Book_ID: 101, Book_Name: 'DSA', Book_Author: 'Ram Kumar' },
   { Book_ID: 102, Book_Name: 'DataBase', Book_Author: 'Shyam Kumar' },
   { Book_ID: 103, Book_Name: 'C++', Book_Author: 'Alice Poudel' },
   { Book_ID: 104, Book_Name: 'Linear Algebra', Book_Author: 'Pritam Kumar' },
   { Book_ID: 105, Book_Name: 'Statistics', Book_Author: 'Purna Bdr Kc' }
 ];
// Insert data using a loop
for (const book of booksData) {
      db.Books.insertOne(book);
}
```

Output:

```
LibaryManagementDatabase> db.createCollection("Books")
{ ok: 1 }
LibaryManagementDatabase> const booksData = [
...    { Book_ID: 101, Book_Name: 'DSA', Book_Author: 'Ram Kumar' },
...    { Book_ID: 102, Book_Name: 'DataBase', Book_Author: 'Shyam Kumar' },
...    { Book_ID: 103, Book_Name: 'C++', Book_Author: 'Alice Poudel' },
...    { Book_ID: 104, Book_Name: 'Linear Algebra', Book_Author: 'Pritam Kumar' },
...    { Book_ID: 105, Book_Name: 'Statistics', Book_Author: 'Purna Bdr Kc' }
...  ];

LibaryManagementDatabase>

LibaryManagementDatabase>    // Insert data using a loop

LibaryManagementDatabase>    for (const book of booksData) {
...      db.Books.insertOne(book);
... }
{
  acknowledged: true,
  insertedId: ObjectId("658b33ffa09e7459e34ef9b2")
}
```

Creating Collection Books:

```
db.createCollection("Journals");

// Function to insert a journal document
function addJournal(Journal_ID, Library_ID, Journal_Author, Journal_Name)
{
  db.Journals.insertOne({
    Journal_ID: Journal_ID,
    Library_ID: Library_ID,
    Journal_Author: Journal_Author,
    Journal_Name: Journal_Name
  });
}

// Adding The Journals
var Library_ID = 5;
var Journal_Details = [
  'Pratiba Patel|Web Development',
  'Alice Pratap|Face Recognition',
  'Sushma Kiran|AI',
  'Dina Nath|ChatBot'
];

// Loop through the journal details and add each journal
for (var i = 0; i < Journal_Details.length; i++) {
  // Split the journal details into author and title
  var details = Journal_Details[i].split('|');
  var Journal_Author = details[0];
  var Journal_Name = details[1];

  // Specify the Journal_ID (replace with your desired value)
   var Journal_ID = 200 + i; // Replace with your desired logic for
Journal_ID

  // Call the addJournal function with the specified Journal_ID
  addJournal(Journal_ID, Library_ID, Journal_Author, Journal_Name);
}
```

Output:

```
LibaryManagementDatabase> var Library_ID = 5;
LibaryManagementDatabase> var Journal_Details = [
...   'Pratiba Patel|Web Development',
...   'Alice Pratap|Face Recognition',
...   'Sushma Kiran|AI',
...   'Dina Nath|ChatBot'
... ];

LibaryManagementDatabase>

LibaryManagementDatabase> // Loop through the journal details and add each
journal

LibaryManagementDatabase> for (var i = 0; i < Journal_Details.length; i++)
{
...   // Split the journal details into author and title
...   var details = Journal_Details[i].split('|');
...   var Journal_Author = details[0];
...   var Journal_Name = details[1];
...
...   // Specify the Journal_ID (replace with your desired value)
...    var Journal_ID = 200 + i; // Replace with your desired logic for
Journal_ID
...
...   // Call the addJournal function with the specified Journal_ID
...   addJournal(Journal_ID, Library_ID, Journal_Author, Journal_Name);
... }

LibaryManagementDatabase>
```

Creating Collection LibraryMembersDetails:

```
db.createCollection("LibraryMembersDetails");

// Insert documents into the MongoDB collection
db.LibraryMembersDetails.insertMany([
  {
    Member_Name: 'Ram Pratap',
    Address: 'Kathmandu',
    Joined_date: new Date(),
    Member_Details: {
      Member_ID: 1,
      Member_Role: 'Teacher',
      Department: 'IT',
      Mobile_number: 9843215987
    }
  },
  {
    Member_Name: 'Bikas Nath',
    Address: 'Birgunj',
    Joined_date: new Date(),
    Member_Details: {
      Member_ID: 2,
      Member_Role: 'Student',
      Department: 'IT',
      Mobile_number: 9843215887
    }
  },
  {
    Member_Name: 'Sita Nuepane',
    Address: 'Sagarmatha',
    Joined_date: new Date(),
    Member_Details: {
      Member_ID: 3,
      Member_Role: 'Student',
      Department: 'Engineering',
      Mobile_number: 9843217987
    }
  },
  {
    Member_Name: 'Prabesh Bhujel',
    Address: 'Kathmandu',
    Joined_date: new Date(),
    Member_Details: {
      Member_ID: 4,
      Member_Role: 'Teacher',
      Department: 'Engineering',
      Mobile_number: 9847215987
    }
  },
  {
    Member_Name: 'Shyam Kumar',
    Address: 'Pokhara',
    Joined_date: new Date(),
    Member_Details: {
      Member_ID: 5,
      Member_Role: 'Teacher',
      Department: 'IT',
      Mobile_number: 98437715987
    }
  }
]);
```

Outputs:

```
acknowledged: true,
insertedIds: {
  '0': ObjectId("658b3827a09e7459e34ef9b7"),
  '1': ObjectId("658b3827a09e7459e34ef9b8"),
  '2': ObjectId("658b3827a09e7459e34ef9b9"),
  '3': ObjectId("658b3827a09e7459e34ef9ba"),
  '4': ObjectId("658b3827a09e7459e34ef9bb")
}
}
```

Creating Collection Books_IssuedandReturn_Return and Books_Fine_Amount:

```
db.createCollection(“Books_Issuedandreturn_Records”)
db.createCollection("Books_Fine_Amount")
// Function to calculate the difference in days between two dates
function calculateDaysDifference(start, end) {
    return (end - start) / (1000 * 60 * 60 * 24); // milliseconds in a day
  }

  // Array of documents to insert
  var documentToInsertForBook_Issuedandreturn_Records = [
    {
      "TRANCTION_BOOK_ID": 1,
      "ISSUEDER_ID": 1,
      "ISSUEDER_NAME": "Ram Kumar",
      "BOOK_ID": 101,
      "BOOK_NAME": "DSA",
      "ISSUED_DATE": ISODate("2022-09-01T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-21T00:00:00.000Z")
    },
    {
      "TRANCTION_BOOK_ID": 2,
      "ISSUEDER_ID": 2,
      "ISSUEDER_NAME": "Bikas Nath",
      "BOOK_ID": 103,
      "BOOK_NAME": "C++",
```

```
      "ISSUED_DATE": ISODate("2022-09-06T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-26T00:00:00.000Z")
    },
    {
      "TRANCTION_BOOK_ID": 3,
      "ISSUEDER_ID": 3,
      "ISSUEDER_NAME": "Sita Nuepane",
      "BOOK_ID": 102,
      "BOOK_NAME": "DataBase",
      "ISSUED_DATE": ISODate("2022-09-07T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-08T00:00:00.000Z")
    },
    {
      "TRANCTION_BOOK_ID": 4,
      "ISSUEDER_ID": 4,
      "ISSUEDER_NAME": "Prabesh Bhujel",
      "BOOK_ID": 105,
      "BOOK_NAME": "Statistics",
      "ISSUED_DATE": ISODate("2022-09-09T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2023-01-21T00:00:00.000Z")
    }
  ];

  // Loop through each document to calculate Day_Issued and FINE_AMOUNT
          documentToInsertForBook_Issuedandreturn_Records.forEach(function
(document) {
       document.Day_Issued  =  calculateDaysDifference(document.ISSUED_DATE,
document.RETRUN_DATE);

    // Calculate FINE_AMOUNT based on Day_Issued
    if (document.Day_Issued > 10 && document.Day_Issued <= 20) {
      document.FINE_AMOUNT = 50;
    } else if (document.Day_Issued > 20 && document.Day_Issued <= 30) {
      document.FINE_AMOUNT = 100;
    } else {
      document.FINE_AMOUNT = 500;
    }

    // Insert each document into the respective collections
    db.Books_Issuedandreturn_Records.insertOne(document);

    db.Books_Fine_Amount.insertOne({
      "TRANCTION_BOOK_ID": document.TRANCTION_BOOK_ID,
      "ISSUEDER_ID":document.ISSUEDER_ID,
      "FINE_AMOUNT": document.FINE_AMOUNT
    });
  });
```

Output:

```
LibaryManagementDatabase>
db.createCollection("Books_Issuedandreturn_Records")
{ ok: 1 }
LibaryManagementDatabase> db.createCollection("Books_Fine_Amount")
{ ok: 1 }
LibaryManagementDatabase> function calculateDaysDifference(start, end) {
...       return (end - start) / (1000 * 60 * 60 * 24); // milliseconds in a
day
...     }
[Function: calculateDaysDifference]
LibaryManagementDatabase>                                                var
documentToInsertForBook_Issuedandreturn_Records = [
...       {
...         "TRANCTION_BOOK_ID": 1,
...         "ISSUEDER_ID": 1,
...         "ISSUEDER_NAME": "Ram Kumar",
...         "BOOK_ID": 101,
...         "BOOK_NAME": "DSA",
...         "ISSUED_DATE": ISODate("2022-09-01T12:00:00.000Z"),
...         "ISSUED_TYPE_QUANTITY": 1,
...         "RETRUN_DATE": ISODate("2022-09-21T00:00:00.000Z")
...       },
...       {
...         "TRANCTION_BOOK_ID": 2,
...         "ISSUEDER_ID": 2,
...         "ISSUEDER_NAME": "Bikas Nath",
...         "BOOK_ID": 103,
...         "BOOK_NAME": "C++",
...         "ISSUED_DATE": ISODate("2022-09-06T12:00:00.000Z"),
...         "ISSUED_TYPE_QUANTITY": 1,
...         "RETRUN_DATE": ISODate("2022-09-26T00:00:00.000Z")
...       },
...       {
...         "TRANCTION_BOOK_ID": 3,
...         "ISSUEDER_ID": 3,
...         "ISSUEDER_NAME": "Sita Nuepane",
...         "BOOK_ID": 102,
...         "BOOK_NAME": "DataBase",
...         "ISSUED_DATE": ISODate("2022-09-07T12:00:00.000Z"),
...         "ISSUED_TYPE_QUANTITY": 1,
...         "RETRUN_DATE": ISODate("2022-09-08T00:00:00.000Z")
...       },
...       {
...         "TRANCTION_BOOK_ID": 4,
...         "ISSUEDER_ID": 4,
...         "ISSUEDER_NAME": "Prabesh Bhujel",
...         "BOOK_ID": 105,
...         "BOOK_NAME": "Statistics",
...         "ISSUED_DATE": ISODate("2022-09-09T12:00:00.000Z"),
...         "ISSUED_TYPE_QUANTITY": 1,
...         "RETRUN_DATE": ISODate("2023-01-21T00:00:00.000Z")
...       }
...     ];

LibaryManagementDatabase>
documentToInsertForBook_Issuedandreturn_Records.forEach(function
(document) {
...                                             document.Day_Issued         =
calculateDaysDifference(document.ISSUED_DATE, document.RETRUN_DATE);
```

```
...
...     // Calculate FINE_AMOUNT based on Day_Issued
...     if (document.Day_Issued > 10 && document.Day_Issued <= 20) {
.....       document.FINE_AMOUNT = 50;
.....       } else if (document.Day_Issued > 20 && document.Day_Issued <=
30) {
.....       document.FINE_AMOUNT = 100;
.....     } else {
.....       document.FINE_AMOUNT = 500;
.....     }
...
...
...  db.Books_Issuedandreturn_Records.insertOne(document);
...
...     db.Books_Fine_Amount.insertOne({
...       "TRANCTION_BOOK_ID": document.TRANCTION_BOOK_ID,
...       "ISSUEDER_ID":document.ISSUEDER_ID,
...       "FINE_AMOUNT": document.FINE_AMOUNT
...     });
...   });

LibaryManagementDatabase>
```

Creating Collections Journalss_IssuedandReturn_Return and Journal_Fine_Amount:

Query:

```
db.createCollection("Journals_Issuedandreturn_Records");
db.createCollection("Journals_Fine_Amount");


  // Insert document into Journals_Issuedandreturn_Records
 // Define the documents to insert
var documentsToInsertForJournals_Issuedandreturn_Records = [
    {
      "TRANCTION_JOURNAL_ID": 1,
      "ISSUEDER_ID": 1,
      "ISSUEDER_NAME": "Ram Pratap",
      "JOURNAL_ID": 201,
      "JOURNAL_NAME": "Web Development",
      "ISSUED_DATE": ISODate("2022-09-01T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-21T00:00:00.000Z")
    },
    {
      "TRANCTION_JOURNAL_ID": 2,
      "ISSUEDER_ID": 3,
      "ISSUEDER_NAME": "Sita Nuepane",
      "JOURNAL_ID": 203,
      "JOURNAL_NAME": "AI",
      "ISSUED_DATE": ISODate("2022-09-03T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-05T00:00:00.000Z")
    },
    {
      "TRANCTION_JOURNAL_ID": 3,
      "ISSUEDER_ID": 2,
      "ISSUEDER_NAME": "Bikas Nath",
      "JOURNAL_ID": 204,
      "JOURNAL_NAME": "ChatBot",
      "ISSUED_DATE": ISODate("2022-09-10T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2022-09-30T00:00:00.000Z")
    },
    {
      "TRANCTION_JOURNAL_ID": 4,
      "ISSUEDER_ID": 4,
      "ISSUEDER_NAME": "Prabesh Bhujel",
      "JOURNAL_ID": 202,
      "JOURNAL_NAME": "Face Recognition",
      "ISSUED_DATE": ISODate("2022-09-15T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2023-01-16T00:00:00.000Z")
    },
    {
      "TRANCTION_JOURNAL_ID": 5,
      "ISSUEDER_ID": 5,
```

```
      "ISSUEDER_NAME": "Shyam Kumar",
      "JOURNAL_ID": 204,
      "JOURNAL_NAME": "ChatBot",
      "ISSUED_DATE": ISODate("2022-09-15T12:00:00.000Z"),
      "ISSUED_TYPE_QUANTITY": 1,
      "RETRUN_DATE": ISODate("2023-01-02T00:00:00.000Z")
    }
  ];

  // Loop through each document to calculate Day_Issued and FINE_AMOUNT
      documentsToInsertForJournals_Issuedandreturn_Records.forEach(function
(document) {
    // Calculate Day_Issued
       document.Day_Issued  =  calculateDaysDifference(document.ISSUED_DATE,
document.RETRUN_DATE);

    // Calculate FINE_AMOUNT based on Day_Issued
    if (document.Day_Issued > 10 && document.Day_Issued <= 20) {
      document.FINE_AMOUNT = 100;
    } else if (document.Day_Issued > 20 && document.Day_Issued <= 30) {
      document.FINE_AMOUNT = 200;
    } else {
      document.FINE_AMOUNT = 800;
    }
  });

    //  Insert  multiple  documents  into  Journals_Issuedandreturn_Records
collection

db.Journals_Issuedandreturn_Records.insertMany(documentsToInsertForJournal
s_Issuedandreturn_Records);

  // Insert corresponding values into Journals_Fine_Amount collection
                    var          documentsToInsertForFineAmount          =
documentsToInsertForJournals_Issuedandreturn_Records.map(function
(document) {
    return {
      "TRANCTION_JOURNAL_ID": document.TRANCTION_JOURNAL_ID,
      "ISSUEDER_ID":document.ISSUEDER_ID,
      "FINE_AMOUNT": document.FINE_AMOUNT
    };
  });

  db.Journals_Fine_Amount.insertMany(documentsToInsertForFineAmount);
```

Output:

```
LibaryManagementDatabase>   // Insert multiple documents into Journals_Issuedandreturn_Records collection

LibaryManagementDatabase>
db.Journals_Issuedandreturn_Records.insertMany(documentsToInsertForJournals_Issuedandreturn_Records);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("658b9859a09e7459e34ef9cc"),
    '1': ObjectId("658b9859a09e7459e34ef9cd"),
    '2': ObjectId("658b9859a09e7459e34ef9ce"),
    '3': ObjectId("658b9859a09e7459e34ef9cf"),
    '4': ObjectId("658b9859a09e7459e34ef9d0")
  }
}
LibaryManagementDatabase>

LibaryManagementDatabase>   // Insert corresponding values into Journals_Fine_Amount collection

LibaryManagementDatabase>                                  var            documentsToInsertForFineAmount            =
documentsToInsertForJournals_Issuedandreturn_Records.map(function (document) {
...     return {
.....       "TRANCTION_JOURNAL_ID": document.TRANCTION_JOURNAL_ID,
.....       "ISSUEDER_ID":document.ISSUEDER_ID,
.....       "FINE_AMOUNT": document.FINE_AMOUNT
.....     };
...   });

LibaryManagementDatabase>

LibaryManagementDatabase>   db.Journals_Fine_Amount.insertMany(documentsToInsertForFineAmount);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("658b9859a09e7459e34ef9d1"),
    '1': ObjectId("658b9859a09e7459e34ef9d2"),
    '2': ObjectId("658b9859a09e7459e34ef9d3"),
    '3': ObjectId("658b9859a09e7459e34ef9d4"),
    '4': ObjectId("658b9859a09e7459e34ef9d5")
  }
}
```

In the provided NOSQL code, the "use LibaryManagementDatabase" command sets the database context to "LibaryManagementDatabase." Subsequently, the code checks for existing collections with the same names using the "drop()" and "find()" functions, ensuring a clean slate for the new data.

The "insertOne()" and "insertMany()" functions are then employed to populate collections such as "Library," "Books," "Journals," and "LibraryMembersDetails" with relevant data. For instance, "Books" and "Journals" collections are populated using loops, and member details are inserted into the "LibraryMembersDetails" collection.

Additionally, the code creates and populates collections like "Books_Issuedandreturn_Records" and "Journals_Issuedandreturn_Records" with transactional data, including details about issued and returned items, issuance dates, and return dates.

Furthermore, fine amounts for books and journals are calculated based on the difference in days between issuance and return, with corresponding values inserted into "Books_Fine_Amount" and "Journals_Fine_Amount" collections.

In summary, this comprehensive NOSQL script establishes a database, creates collections, and systematically inserts data, reflecting a structured library management system.

Comparison of Scripts

The comparison between Oracle SQL query and MongoDB is given below:

TASK FOUR: JOIN

Query 1 -A join of three or More tables With restrictions.

The first query is a SQL join query that selects columns from four tables: BOOKS_ISSUEDANDRETURN_RECORDS, LIBRARY_MEMBERS_DETAILS_TABLES,BOOKS and BOOKS_FINEAMOUNT.It uses left join, right join, and left join operators to join the tables and a where clause to filter results.

The Second query is a MongoDB aggregation query that uses $lookup operators to perform joins between the BOOKS_ISSUEDANDRETURN_RECORDS,BOOKS, LIBRARY_MEMBERS_DETAILS_TABLES and BOOKS_FINEAMOUNT.It includes a $match operator to filter results and a $project operator to select fields.
The goal of both queries is to retrieve information about Books, LibraryMembers whose Fine_Amount is greater than 50.

SQL Query	MongoDB Script

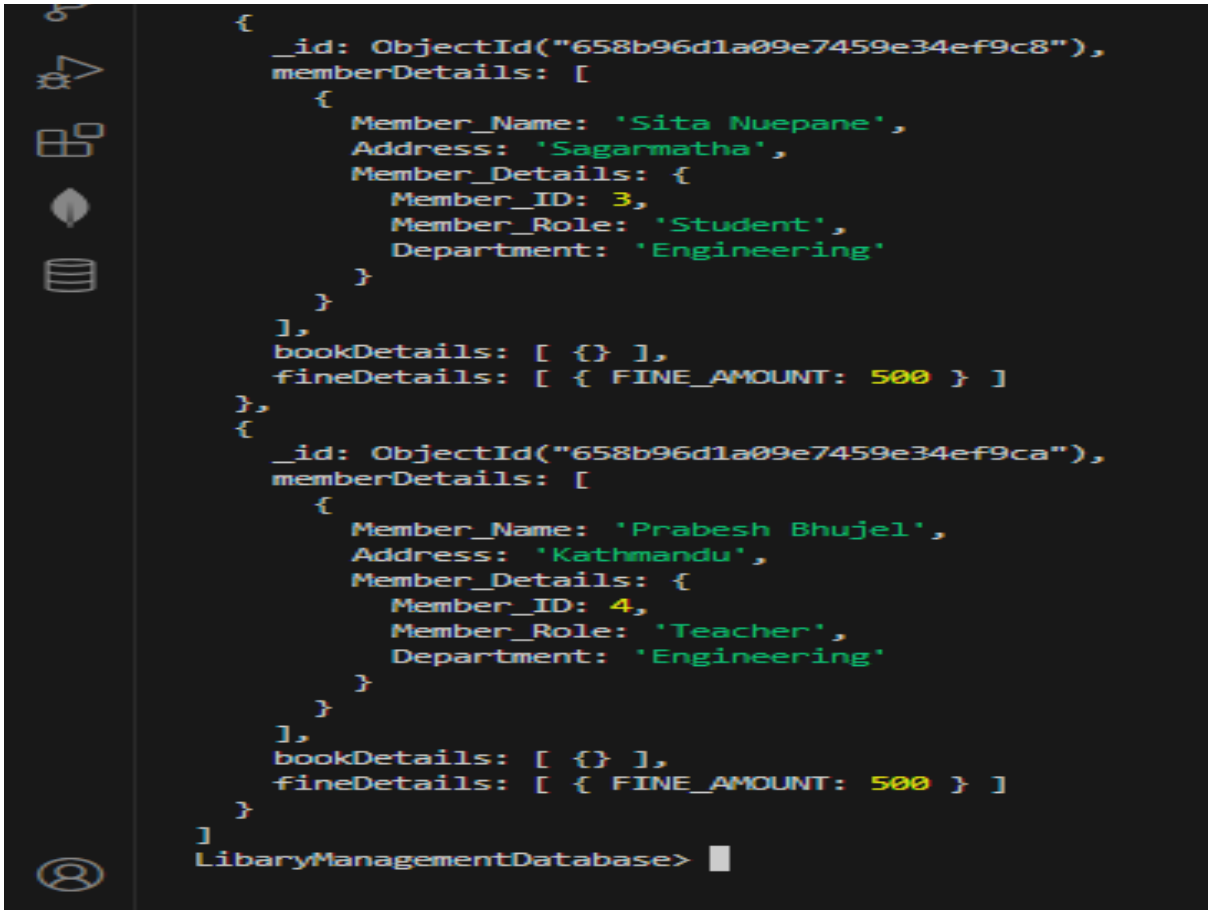
SELECT
Library_Members_Details_Tables.MEMBER_NAME,
Library_Members_Details_Tables.ADDRESS,
Library_Members_Details_Tables.MEMBER_ID,
Library_Members_Details_Tables.MEMBER_ROLE,
Library_Members_Details_Tables.DEPARTMENT,
BOOKS_ISSUEDANDRETURN_RECORDS.BOOK_ID,
BOOKS_ISSUEDANDRETURN_RECORDS.BOOK_NAME,
BOOKS_ISSUEDANDRETURN_RECORDS.DAYS_ISSUED,
BOOKS_FINEAMOUNT.FINE_AMOUNT
FROM BOOKS_ISSUEDANDRETURN_RECORDS
 	
LEFT JOIN Library_Members_Details_Tables on BOOKS_ISSUEDANDRETURN_RECORDS.ISSUEDER_ID = Library_Members_Details_Tables.Member_ID
 	
RIGHT JOIN Books on BOOKS_ISSUEDANDRETURN_RECORDS.Book_ID = Books.Book_ID
 	
LEFT JOIN BOOKS_FINEAMOUNT ON BOOKS_ISSUEDANDRETURN_RECORDS.TRANCTION_BOOK_ID = BOOKS_FINEAMOUNT.TRANCTION_BOOK_ID
 	
WHERE BOOKS_FINEAMOUNT.FINE_AMOUNT>50 ;
	db.Books_Issuedandreturn_Records.aggregate([
	{
	$lookup: {
	from: "LibraryMembersDetails",
	localField: "ISSUEDER_ID",
	foreignField: "Member_Details.Member_ID",
	as: "memberDetails"
	}
	},
	{
	$lookup: {
	from: "Books",
	localField: "BOOK_ID",
	foreignField: "Book_ID",
	as: "bookDetails"
	}
	},
	{
	$lookup: {
	from: "Books_Fine_Amount",
	localField: "TRANCTION_BOOK_ID",
	foreignField: "TRANCTION_BOOK_ID",
	as: "fineDetails"
	}
	},
	{
	$match: {
	"fineDetails.FINE_AMOUNT": { $gt: 50 }
	}
	},
	{
	$project: {
	"memberDetails.Member_Name": 1,
	"memberDetails.Address": 1,

SQL Query	MongoDB Script
	"memberDetails.Member_Details.Member_ID": 1, "memberDetails.Member_Details.Member_Role": 1, "memberDetails.Member_Details.Department": 1, "bookDetails.BOOK_ID": 1, "bookDetails.BOOK_NAME": 1, "DAYS_ISSUED": 1, "fineDetails.FINE_AMOUNT": 1 } }]);

Oracle Output:

MEMBER_NAME	ADDRESS	MEMBER_ID	MEMBER_ROLE	DEPARTMENT	BOOK_ID	BOOK_NAME	DAYS_ISSUED	FINE_AMOUNT
Sita Nuepane	Sagarmatha	3	Student	Engineering	102	DataBase	0	500
Prabesh Bhujel	Kathmandu	4	Teacher	Engineering	105	Statistics	133	500

Mongodb Output:



TASK FIVE : UNION

Query 2: A query which uses one (or more) of UNION,DIFFERENCE or INTERSECTION
Both queries retrieve the data of Book,Journal,Liberal Member details who has the Fine.The first query is a SQL Union query that selects all columns from the tables BOOKS,JOURNALS,BOOKS_ISSUEDANDRETURN_RECORDS,JOURNALS_ISSUEDANDRETURN_RECORDS,JOURNALS_FINEAMOUNT,BOOKS_FINEAMOUNT joining them . It uses the UNION keyword to combine the results of two separate queries.
The second query is a Mongodb aggregation query that performs a join between the LIBRARY_MEMBERS_DETAILS_TABLES with BOOKS,BOOKS_ISSUEDANDRETURN_RECORDS and BOOKS_FINEAMOUNT collections using the $lookup operator. It then use the $ unionWith operator to UNION JOURNALS,JOURNALS_ISSUEDANDRETURN_RECORDS,JOURNALSS_FINEAMOUNT.
The goal of both queries is to retrieve information about Readers of both Books and Journals who has minimal fine.

SQL Query	MongoDB Script

SELECT
Library_Members_Details_Tables.MEMBER_NAME,
Library_Members_Details_Tables.ADDRESS,
Library_Members_Details_Tables.MEMBER_ID,
Library_Members_Details_Tables.MEMBER_ROLE,
Library_Members_Details_Tables.DEPARTMENT,
BOOKS_ISSUEDANDRETURN_RECORDS.BOOK_ID,
BOOKS_ISSUEDANDRETURN_RECORDS.BOOK_NAME,
BOOKS_ISSUEDANDRETURN_RECORDS.DAYS_ISSUED,
BOOKS_FINEAMOUNT.FINE_AMOUNT
 	
FROM BOOKS_ISSUEDANDRETURN_RECORDS
 	
LEFT JOIN Library_Members_Details_Tables on BOOKS_ISSUEDANDRETURN_RECORDS.ISSUEDER_ID = Library_Members_Details_Tables.Member_ID
 	
RIGHT JOIN Books on BOOKS_ISSUEDANDRETURN_RECORDS.Book_ID = Books.Book_ID
 	
lEFT JOIN BOOKS_FINEAMOUNT ON BOOKS_ISSUEDANDRETURN_RECORDS.TRANCTION_BOOK_ID = BOOKS_FINEAMOUNT.TRANCTION_BOOK_ID
WHERE BOOKS_FINEAMOUNT.FINE_AMOUNT>50
 	
UNION
 	
SELECT
Library_Members_Details_Tables.MEMBER_NAME,
Library_Members_Details_Tables.ADDRESS,
Library_Members_Details_Tables.MEMBER_ID,
Library_Members_Details_Tables.MEMBER_ROLE,
Library_Members_Details_Tables.DEPARTMENT,
 	
JOURNALS_ISSUEDANDRETURN_RECORDS.JOURNAL_ID,
JOURNALS_ISSUEDANDRETURN_RECORDS.JOURNAL_NAME,
JOURNALS_ISSUEDANDRETURN_RECORDS.DAYS_ISSUED,
JOURNALS_FINEAMOUNT.FINE_AMOUNT
 	
FROM JOURNALS_ISSUEDANDRETURN_RECORDS
 	
LEFT JOIN Library_Members_Details_Tables on JOURNALS_ISSUEDANDRETURN_RECORDS.ISSUEDER_ID = Library_Members_Details_Tables.Member_ID
 	
RIGHT JOIN Journals on JOURNALS_ISSUEDANDRETURN_RECORDS.Journal_ID =
	db.Books_Issuedandreturn_Records.aggregate([
	{
	$lookup: {
	from: "LibraryMembersDetails",
	localField: "ISSUEDER_ID",
	foreignField: "Member_Details.Member_ID",
	as: "memberDetails"
	}
	},
	{
	$lookup: {
	from: "Books",
	localField: "BOOK_ID",
	foreignField: "Book_ID",
	as: "bookDetails"
	}
	},
	{
	$lookup: {
	from: "Books_Fine_Amount",
	localField: "TRANCTION_BOOK_ID",
	foreignField: "TRANCTION_JOURNAL_ID",
	as: "fineDetails"
	}
	},
	{
	$match: {
	"fineDetails.FINE_AMOUNT": { $gt: 49 }
	}
	},
	{
	$project: {
	"memberDetails.Member_Name": 1,
	"memberDetails.Address": 1,
	
	"memberDetails.Member_Details.Member_ID": 1,
	
	"memberDetails.Member_Details.Member_Role": 1,
	
	"memberDetails.Member_Details.Department": 1,
	
	"BOOKS_ISSUEDANDRETURN_RECORDS.BOOK_ID": 1,
	
	"BOOKS_ISSUEDANDRETURN_RECOR

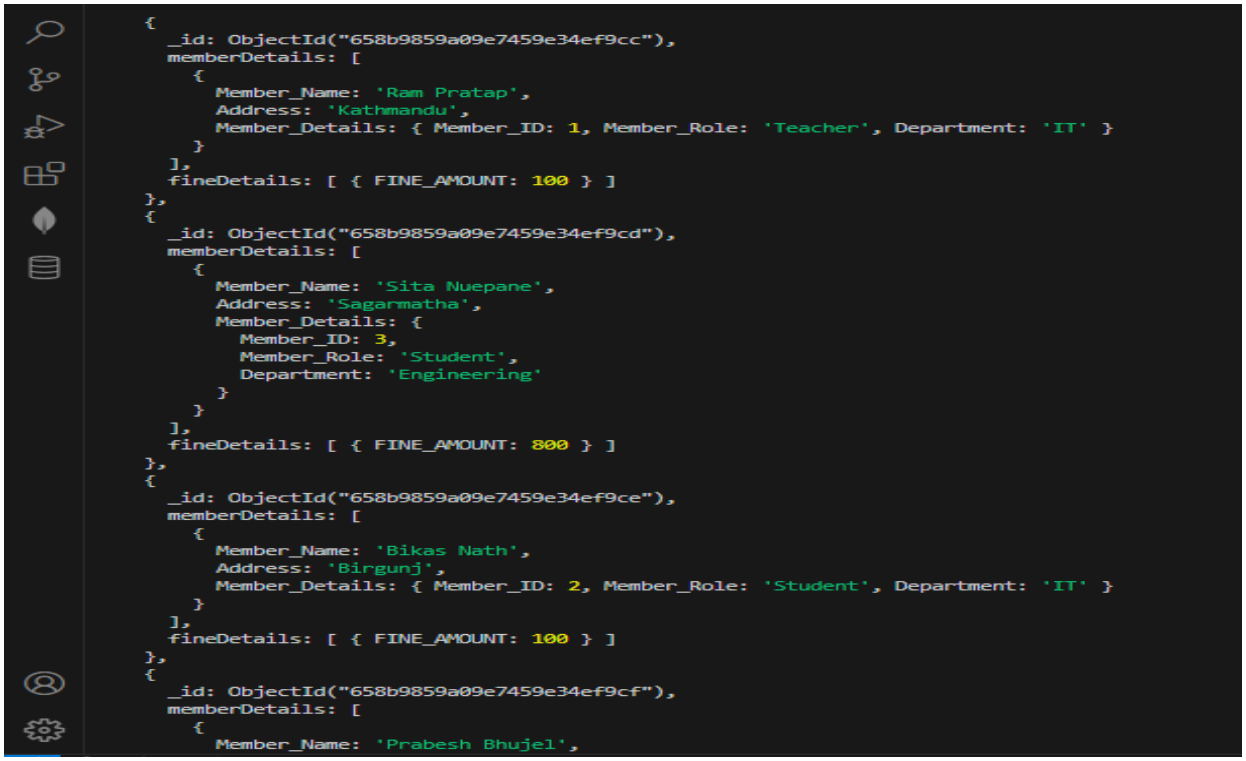
Journals.Journal_ID
 	
lEFT JOIN JOURNALS_FINEAMOUNT ON JOURNALS_ISSUEDANDRETURN_RECORDS.TRANCTION_JOURNAL_ID = JOURNALS_FINEAMOUNT.TRANCTION_JOURNAL_ID
 	
WHERE JOURNALS_FINEAMOUNT.FINE_AMOUNT>=100 ;
	DS.BOOK_NAME": 1,
	
	"BOOKS_ISSUEDANDRETURN_RECORDS.DAYS_ISSUED": 1,
	"fineDetails.FINE_AMOUNT": 1
	}
	},
	{
	$unionWith: {
	coll: "Journals_Issuedandreturn_Records",
	pipeline: [
	{
	$lookup: {
	from: "LibraryMembersDetails",
	localField: "ISSUEDER_ID",
	foreignField: "Member_Details.Member_ID",
	as: "memberDetails"
	}
	},
	{
	$lookup: {
	from: "Journals",
	localField: "JOURNAL_ID",
	foreignField: "Journal_ID",
	as: "journalDetails"
	}
	},
	{
	$lookup: {
	from: "Journals_Fine_Amount",
	localField: "TRANCTION_JOURNAL_ID",
	foreignField: "TRANCTION_JOURNAL_ID",
	as: "fineDetails"
	}
	},
	{
	$match: {
	"fineDetails.FINE_AMOUNT": { $gte: 100 }
	}
	},
	{
	$project: {
	"memberDetails.Member_Name": 1,
	"memberDetails.Address": 1,
	
	"memberDetails.Member_Details.Member_ID": 1,
	
	"memberDetails.Member_Details.Member_Role": 1,

	"memberDetails.Member_Details.Department": 1, "JOURNALS_ISSUEDANDRETURN_RECORDS.JOURNAL_ID": 1, "JOURNALS_ISSUEDANDRETURN_RECORDS.JOURNAL_NAME": 1, "JOURNALS_ISSUEDANDRETURN_RECORDS.DAYS_ISSUED": 1, "fineDetails.FINE_AMOUNT": 1 } }] } }]);

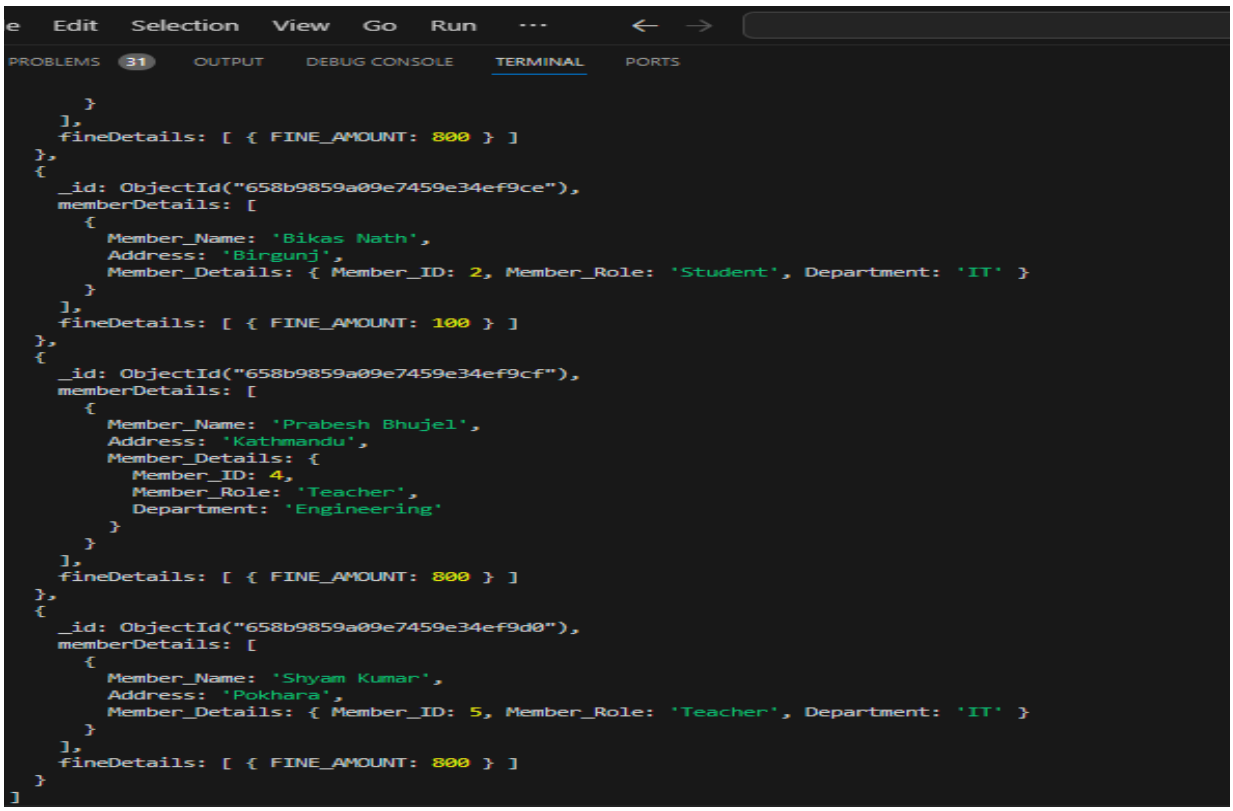
Oracle Output:

MEMBER_NAME	ADDRESS	MEMBER_ID	MEMBER_ROLE	DEPARTMENT	BOOK_ID	BOOK_NAME	DAYS_ISSUED	FINE_AMOUNT
Bikas Nath	Birgunj	2	Student	IT	204	ChatBot	19	100
Prabesh Bhujel	Kathmandu	4	Teacher	Engineering	105	Statistics	133	500
Prabesh Bhujel	Kathmandu	4	Teacher	Engineering	202	Face Recognition	122	800
Ram Pratap	Kathmandu	1	Teacher	IT	201	Web Developemnt	19	100
Shyam Kumar	Pokhara	5	Teacher	IT	204	ChatBot	108	800
Sita Nuepane	Sagarmatha	3	Student	Engineering	102	DataBase	0	500
Sita Nuepane	Sagarmatha	3	Student	Engineering	203	AI	1	800

Mongodb Outputs:



```
    {
      _id: ObjectId("658b9859a09e7459e34ef9cc"),
      memberDetails: [
        {
          Member_Name: 'Ram Pratap',
          Address: 'Kathmandu',
          Member_Details: { Member_ID: 1, Member_Role: 'Teacher', Department: 'IT' }
        }
      ],
      fineDetails: [ { FINE_AMOUNT: 100 } ]
    },
    {
      _id: ObjectId("658b9859a09e7459e34ef9cd"),
      memberDetails: [
        {
          Member_Name: 'Sita Nuepane',
          Address: 'Sagarmatha',
          Member_Details: {
            Member_ID: 3,
            Member_Role: 'Student',
            Department: 'Engineering'
          }
        }
      ],
      fineDetails: [ { FINE_AMOUNT: 800 } ]
    },
    {
      _id: ObjectId("658b9859a09e7459e34ef9ce"),
      memberDetails: [
        {
          Member_Name: 'Bikas Nath',
          Address: 'Birgunj',
          Member_Details: { Member_ID: 2, Member_Role: 'Student', Department: 'IT' }
        }
      ],
      fineDetails: [ { FINE_AMOUNT: 100 } ]
    },
    {
      _id: ObjectId("658b9859a09e7459e34ef9cf"),
      memberDetails: [
        {
          Member_Name: 'Prabesh Bhujel',
```



```
e  Edit  Selection  View  Go  Run  ···

PROBLEMS 31  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

      }
    ],
    fineDetails: [ { FINE_AMOUNT: 800 } ]
  },
  {
    _id: ObjectId("658b9859a09e7459e34ef9ce"),
    memberDetails: [
      {
        Member_Name: 'Bikas Nath',
        Address: 'Birgunj',
        Member_Details: { Member_ID: 2, Member_Role: 'Student', Department: 'IT' }
      }
    ],
    fineDetails: [ { FINE_AMOUNT: 100 } ]
  },
  {
    _id: ObjectId("658b9859a09e7459e34ef9cf"),
    memberDetails: [
      {
        Member_Name: 'Prabesh Bhujel',
        Address: 'Kathmandu',
        Member_Details: {
          Member_ID: 4,
          Member_Role: 'Teacher',
          Department: 'Engineering'
        }
      }
    ],
    fineDetails: [ { FINE_AMOUNT: 800 } ]
  },
  {
    _id: ObjectId("658b9859a09e7459e34ef9d0"),
    memberDetails: [
      {
        Member_Name: 'Shyam Kumar',
        Address: 'Pokhara',
        Member_Details: { Member_ID: 5, Member_Role: 'Teacher', Department: 'IT' }
      }
    ],
    fineDetails: [ { FINE_AMOUNT: 800 } ]
  }
]
```

TASK SIX : NESTED QUERIES

Query 3 : A query which requires use of either a nested table or subtypes .
The first is SQL nested queries.It select Member_Name,Address,Member_Role,Department, Mobile number from Table Library_Members_Details_Tables by matching Member_ID to the Table Book_Fine_Amount Issude_Id whose Fine_Amount >50 using IN method.IN keyword Is used to join two tables making a later table as nested.

The second query is Mongodb script in which two collections LibraryMembersDetails and Books_Fine_Amount collection are join using $lookup ,to filter $match is used and to select the fields $project is used
The goal of two queries is to extract the Member information which is absent in one table or collection by making the next table or collection nested .

Oracle query

```
SELECT
Library_Members_Details_Tables.MEMBER_NAME,
Library_Members_Details_Tables.ADDRESS,
Library_Members_Details_Tables.MEMBER_ROLE,
Library_Members_Details_Tables.DEPARTMENT,
Library_Members_Details_Tables.MOBILE_NUMBER

FROM Library_Members_Details_Tables

WHERE Library_Members_Details_Tables.MEMBER_ID IN (
   SELECT BOOKS_FINEAMOUNT.ISSUEDER_ID
   FROM BOOKS_FINEAMOUNT
   WHERE BOOKS_FINEAMOUNT.FINE_AMOUNT > 50
);
```

Mongodb Scripts

```
db.LibraryMembersDetails.aggregate([
   {
    $lookup: {
      from: "Books_Fine_Amount",
      localField:
"Member_Details.Member_ID",
      foreignField: "ISSUEDER_ID",
      as: "fineDetails"
    }
   },
   {
    $match: {
      "fineDetails.FINE_AMOUNT": {
$gt: 50 }
    }
   },
   {
    $project: {
      "Member_Name": 1,
      "Address": 1,
      "Member_Details.Member_Role": 1,
      "Member_Details.Department": 1,
      "Member_Details.Mobile_number":
1,
      "_id": 0
    }
   }
 ]);
```

Oracle Output:

MEMBER_NAME	ADDRESS	MEMBER_ROLE	DEPARTMENT	MOBILE_NUMBER
Sita Nuepane	Sagarmatha	Student	Engineering	9843217987
Prabesh Bhujel	Kathmandu	Teacher	Engineering	9847215987

Mongodb Output:

```
    {
      Member_Name: 'Sita Nuepane',
      Address: 'Sagarmatha',
      Member_Details: {
        Member_Role: 'Student',
        Department: 'Engineering',
        Mobile_number: 9843217987
      }
    },
    {
      Member_Name: 'Prabesh Bhujel',
      Address: 'Kathmandu',
      Member_Details: {
        Member_Role: 'Teacher',
        Department: 'Engineering',
        Mobile_number: 9847215987
      }
    }
]
LibaryManagementDatabase>
```

TASK SEVEN : TEMPORAL FEATURE QUERY

Query 4 : A query using temporal features (e.g., timestamps, intervals, etc.) of Oracle SQL

The first query is SQL which uses the temporal feature Current_Timestamp.In this section this feature is used to return all members that were joined to the Library in the last year.

The second script is Mongodb.In this script $expr operator is used for aggregation expressions ,$and operator contains two conditions:

$gte to check if 'joined_date' is greater than or equal to one year ago from current date,

$lte to check if 'joined date' is less than the current date

Likewise, Projection is used for selecting fields.

The goal of two queries is to find the member who joined last year.

Oracle query

```
SELECT
   Member_Name,
   Address,
   Member_ID,
   Member_Role,
   Department,
   Joined_date
FROM
   Library_Members_Details_Tables
WHERE
   Joined_date >= CURRENT_TIMESTAMP -
INTERVAL '1' YEAR;
```

Mongodb Script

```
var query = {
   $expr: {
      $and: [
         { $gte: ["$Joined_date", { $subtract: [new
Date(), { $multiply: [365, 24, 60, 60, 1000] }] }] },
         { $lt: ["$Joined_date", new Date()] }
      ]
   }
};

// Projection to include only specific fields in the result
var projection = {
   Member_Name: 1,
   Address: 1,
   Member_ID: 1,
   Member_Role: 1,
   Department: 1,
   Joined_date: 1,
   _id: 0 // Exclude the default _id field if you don't
want it in the result
};

// Execute the find query
var result = db.LibraryMembersDetails.find(query,
projection);

// Display the result
result.forEach(function (doc) {
   printjson(doc);
});
```

Oracle Output:

MEMBER_NAME	ADDRESS	MEMBER_ID	MEMBER_ROLE	DEPARTMENT	JOINED_DATE
Ram Pratap	Kathmandu	1	Teacher	IT	27-DEC-23 04.51.24.596911 AM
Bikas Nath	Birgunj	2	Student	IT	27-DEC-23 04.51.24.603666 AM
Sita Nuepane	Sagarmatha	3	Student	Engineering	27-DEC-23 04.51.24.606861 AM
Prabesh Bhujel	Kathmandu	4	Teacher	Engineering	27-DEC-23 04.51.24.609795 AM
Shyam Kumar	Pokhara	5	Teacher	IT	27-DEC-23 04.51.24.612560 AM

Mongodb Outputs:

```
{
  Member_Name: 'Ram Pratap',
  Address: 'Kathmandu',
  Joined_date: ISODate("2023-12-26T20:31:35.963Z")
}
{
  Member_Name: 'Bikas Nath',
  Address: 'Birgunj',
  Joined_date: ISODate("2023-12-26T20:31:35.963Z")
}
{
  Member_Name: 'Sita Nuepane',
  Address: 'Sagarmatha',
  Joined_date: ISODate("2023-12-26T20:31:35.963Z")
}
{
  Member_Name: 'Prabesh Bhujel',
  Address: 'Kathmandu',
  Joined_date: ISODate("2023-12-26T20:31:35.963Z")
}
{
  Member_Name: 'Shyam Kumar',
  Address: 'Pokhara',
  Joined_date: ISODate("2023-12-26T20:31:35.963Z")
}
```

TASK EIGHT : TEMPORAL FEATURE QUERY

Query 5: A query using OLAP (e.g., ROLLUP, CUBE, PARTITION) features of Oracle SQL

The first Query is SQL which uses the Partition OLAP features.Min,Max,Avg features are used for calculating minimum,average,maximum fine amount along with running total of table Books_Fine_Amount

The second Query is a NOSQL query. Aggregate function is used along with $group to group by Transaction_Book_Id ,likewise $max,$min and $avg is used in a Books_Fine_Amount collection.

The goal of both queries is to find the maximum,minimum and average Fine Amount

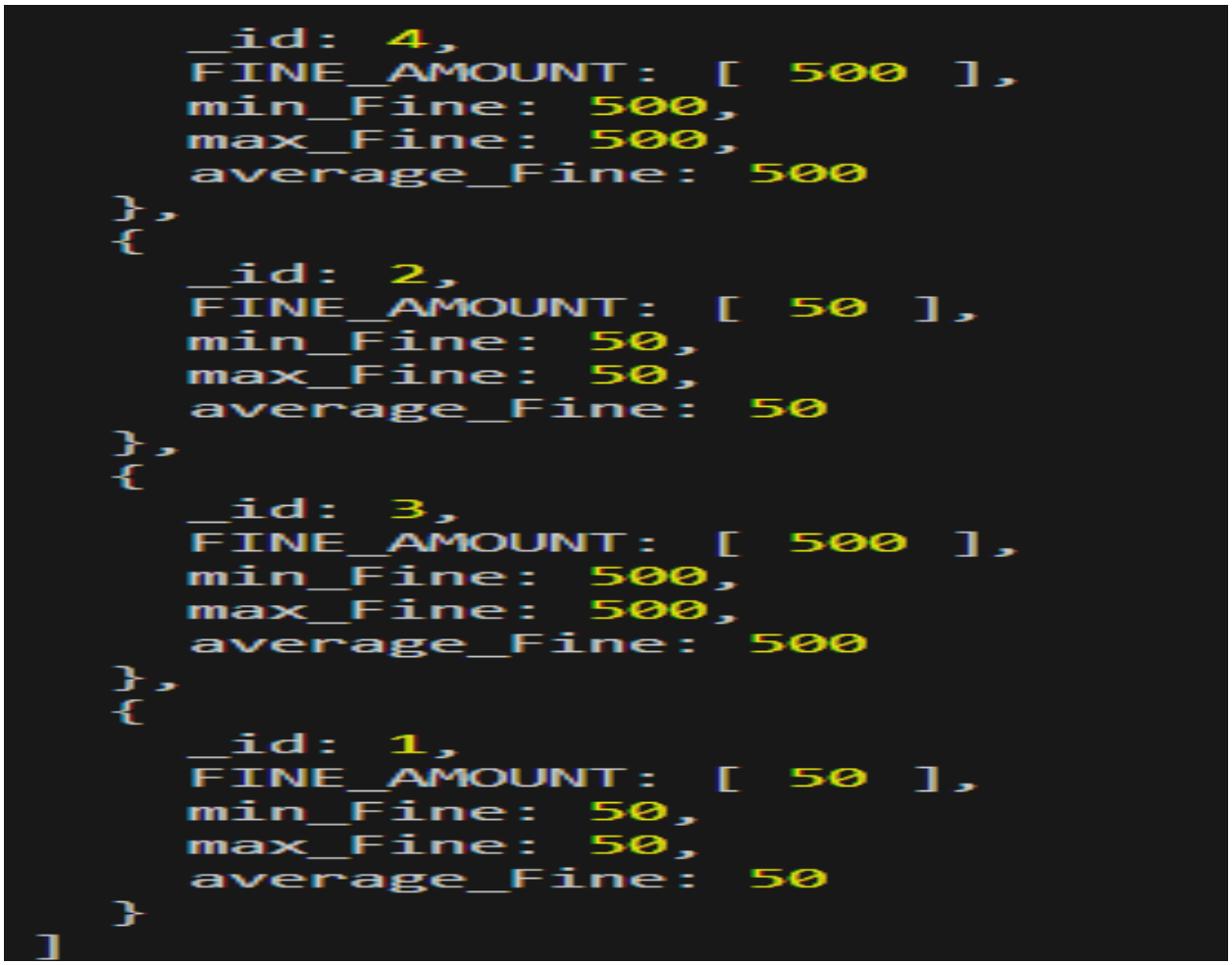
Oracle query	**Mongodb Script**

select TRANCTION_BOOK_ID,
 	
sum(FINE_AMOUNT) over(order by FINE_AMOUNT rows between unbounded preceding and current row) as Runningtotal,
 	
avg(FINE_AMOUNT) over(order by FINE_AMOUNT rows between unbounded preceding and unbounded following) as avg,
 	
min(FINE_AMOUNT) over(order by FINE_AMOUNT rows between unbounded preceding and unbounded following) as min,
 	
max(FINE_AMOUNT) over(order by FINE_AMOUNT rows between unbounded preceding and unbounded following) as max from BOOKS_FINEAMOUNT;
	db.Books_Fine_Amount.aggregate(
	{
	$group: {
	_id:'$TRANCTION_BOOK_ID',
	
	FINE_AMOUNT: { $push: "$FINE_AMOUNT" },
	min_Fine: { $min: '$FINE_AMOUNT' },
	max_Fine: { $max: '$FINE_AMOUNT' },
	average_Fine: { $avg: '$FINE_AMOUNT' }
	}
	}
	)

Oracle Output:

TRANCTION_BOOK_ID	RUNNINGTOTAL	AVG	MIN	MAX
1	50	275	50	500
2	100	275	50	500
3	600	275	50	500
4	1100	275	50	500

Mongodb Output:



Conclusion:

In conclusion, I have successfully created both an Oracle SQL and MongoDB database, and implement five queries on each to evaluate their performance.I performed queries such as joins,union,nested query,To_Timestamp, and OLAP(partations)and made sure that the result produced by both database were similar.

After evaluating the performance of both databases,I found that they were efficient and capable of handling the queries with ease.However, the choice is depends upon the needs and requirements of the project.

For example, if project needs structured data,need to perform ACID properties so in this condition SQL is best but in contrast if project need flexible schema ,rapid development in that situation NOSQL is best.

REFERENCES:

Author(s): Oracle Corporation
Year of publication: 2001 (you can use the last update date if available)
Title of the document: Oracle® Database PL/SQL User's Guide and Reference
URL: https://docs.oracle.com/cd/B10500_01/appdev.920/a96624/toc.htm

MongoDB Documentation
URL: https://www.mongodb.com/docs/

Boicea, A., Radulescu, F. and Agapin, L.I. (2012) "MongoDB vs oracle -- database comparison," 2012 Third International Conference on Emerging Intelligent Data and Web Technologies [Preprint]. Available at:
https://doi.org/10.1109/eidwt.2012.32.