Mini Project: Database Design and Implementation

This document presents the complete mini-project combining both the design and implementation phases of a relational database system for a Public Library scenario. It consolidates all tasks including business scenario, entity identification, normalization, ERD creation, SQL implementation, data insertion, and query execution.

Part 1: Database Design and Normalisation

Business Scenario

In the vibrant city of Geelong, Victoria, the Geelong Regional Library (GRL) serves as a hub of knowledge and community engagement. Its mission is to enrich lives through education, entertainment, and exploration. The library manages members, staff, branches, items such as books and DVDs, loan transactions, and community programs. These interactions form the backbone of the library's operations and establish the need for a robust database system to handle data efficiently.

Entities and Attributes

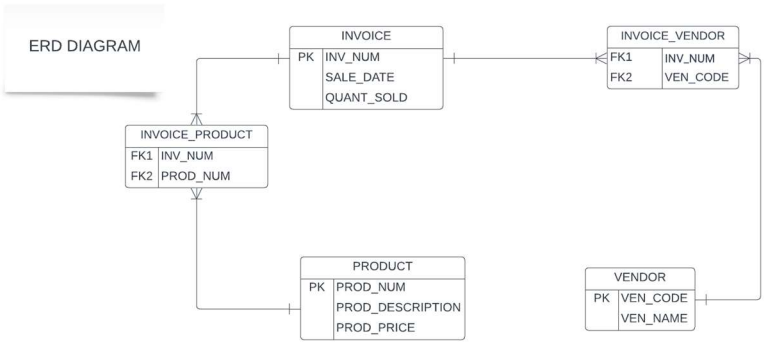
The following entities and their attributes were identified:
- Member: MemberID (PK), FirstName, LastName, Address, Email, PhoneNumber, DateOfBirth, MembershipType, MembershipExpiryDate
- Library Item: ItemID (PK), Title, Author, ISBN, Category, Format, PublicationYear, AvailabilityStatus
- Library Branch: BranchID (PK), BranchName, Address, PhoneNumber, OperatingHours
- Loan Transaction: TransactionID (PK), LoanDate, DueDate, ReturnDate, MemberID (FK), ItemID (FK)
- Library Program: ProgramID (PK), ProgramName, Description, Date, Location, AttendeeCount, BranchID (FK)
- Staff: StaffID (PK), FirstName, LastName, Email, PhoneNumber, BranchID (FK)

Normalization

Entities were normalized up to Third Normal Form (3NF). All entities satisfied 1NF, 2NF, and 3NF. The Loan Transaction entity was refined by separating concerns to ensure no transitive dependencies.

Entity Relationship Diagram (ERD)

An ERD was created showing entities, attributes, primary and foreign keys, and their relationships using Crow's Foot notation.



Part 2: Database Implementation

Table Creation

The following SQL statements were used to create tables:
- CREATE TABLE Member (...)
- CREATE TABLE LibraryItem (...)
- CREATE TABLE LibraryBranch (...)
- CREATE TABLE LoanTransaction (...)
- CREATE TABLE LibraryProgram (...)
- CREATE TABLE Staff (...)

Data Insertion

At least five records were inserted into each table using INSERT statements. For example, the Member table was populated with members such as John Doe, Jane Smith, Michael Johnson, Emily Wilson, and David Lee.

Schema Modification

An additional column 'IsActive' (BOOLEAN DEFAULT TRUE) was added to the Member table. Records were updated so that members whose membership expiry date had passed were set to IsActive = FALSE.

Queries and Results

Several SQL queries were executed to demonstrate functionality:
- INNER JOIN with ORDER BY: Retrieved member names with borrowed item titles.
- WHERE with IN: Retrieved program details for specific branches.
- DATE functions: Calculated member ages from their date of birth.
- VIEW creation: Created a BorrowedItems view joining Member, LoanTransaction, and LibraryItem.

Conclusion

This project successfully designed and implemented a relational database for a public library. The database covers member management, branch operations, loan tracking, staff allocation, and program coordination. Normalization ensured data integrity and efficiency, while SQL implementation validated the design with data operations and queries. The system demonstrates how a database can support community-driven organizations in managing their resources effectively.

Appendix: SQL Scripts and Outputs

Table Creation Scripts

```
CREATE TABLE Member (
    MemberID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Address VARCHAR(255),
    Email VARCHAR(100),
    PhoneNumber VARCHAR(20),
    DateOfBirth DATE,
    MembershipType VARCHAR(50),
    MembershipExpiryDate DATE
);
```

```
CREATE TABLE LibraryItem (
    ItemID INT PRIMARY KEY,
    Title VARCHAR(100),
    Author VARCHAR(100),
    ISBN VARCHAR(20),
    Category VARCHAR(50),
    Format VARCHAR(20),
    PublicationYear INT,
    AvailabilityStatus VARCHAR(50)
);

CREATE TABLE LibraryBranch (
    BranchID INT PRIMARY KEY,
    BranchName VARCHAR(100),
    Address VARCHAR(255),
    PhoneNumber VARCHAR(20),
    OperatingHours VARCHAR(255)
);

CREATE TABLE LoanTransaction (
    TransactionID INT PRIMARY KEY,
    LoanDate DATE,
    DueDate DATE,
    ReturnDate DATE,
    MemberID INT,
    ItemID INT,
    FOREIGN KEY (MemberID) REFERENCES Member(MemberID),
    FOREIGN KEY (ItemID) REFERENCES LibraryItem(ItemID)
);

CREATE TABLE LibraryProgram (
    ProgramID INT PRIMARY KEY,
    ProgramName VARCHAR(100),
    Description TEXT,
    ProgramDate DATE,
    Location VARCHAR(255),
    AttendeeCount INT,
    BranchID INT,
    FOREIGN KEY (BranchID) REFERENCES LibraryBranch(BranchID)
);

CREATE TABLE Staff (
    StaffID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    PhoneNumber VARCHAR(20),
    BranchID INT,
    FOREIGN KEY (BranchID) REFERENCES LibraryBranch(BranchID)
);
```

Sample Data Inserts

```
INSERT INTO Member (MemberID, FirstName, LastName, Address, Email, PhoneNumber, DateOfBirth,
MembershipType, MembershipExpiryDate)
VALUES
(1, 'John', 'Doe', '123 Main St', 'john@example.com', '555-123-4567', '1990-05-15',
'Regular', '2024-05-15'),
(2, 'Jane', 'Smith', '456 Oak St', 'jane@example.com', '555-987-6543', '1985-09-22',
```

```
'Premium', '2023-08-30'),
(3, 'Michael', 'Johnson', '789 Elm St', 'michael@example.com', '555-789-1234', '2000-02-10',
'Regular', '2024-04-12'),
(4, 'Emily', 'Wilson', '321 Birch St', 'emily@example.com', '555-567-8901', '1998-11-03',
'Premium', '2023-09-25'),
(5, 'David', 'Lee', '987 Cedar St', 'david@example.com', '555-234-5678', '1987-07-20',
'Regular', '2024-03-05');
```

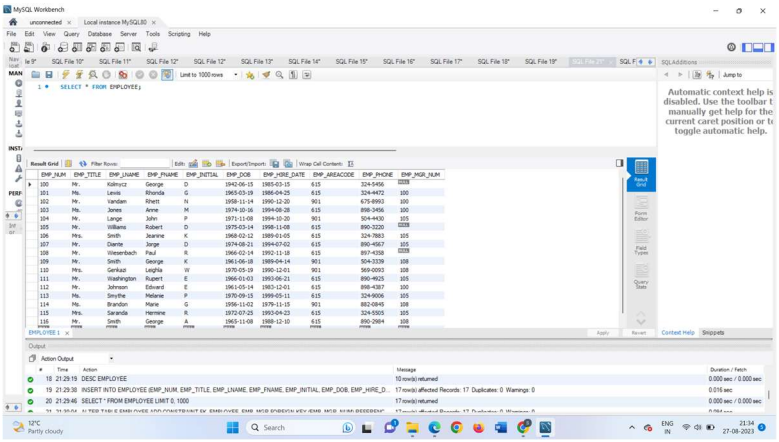
Schema Modification

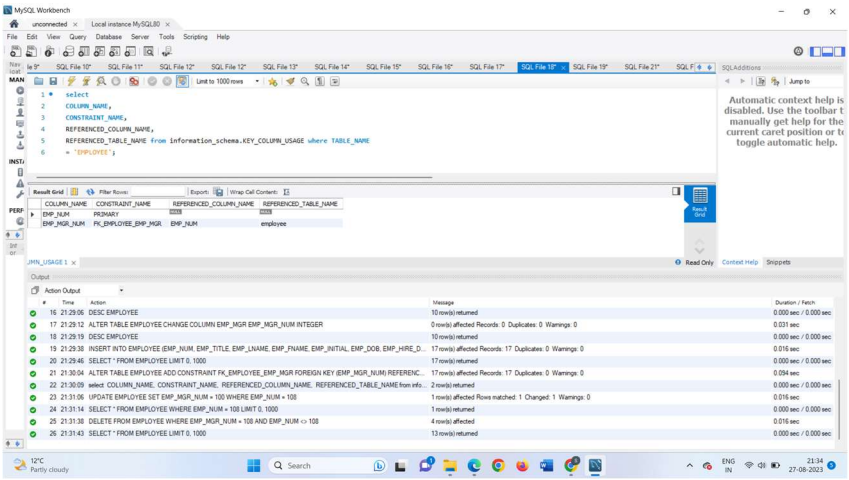
```
ALTER TABLE Member
ADD COLUMN IsActive BOOLEAN DEFAULT TRUE;

UPDATE Member
SET IsActive = FALSE
WHERE MembershipExpiryDate < CURRENT_DATE;
```

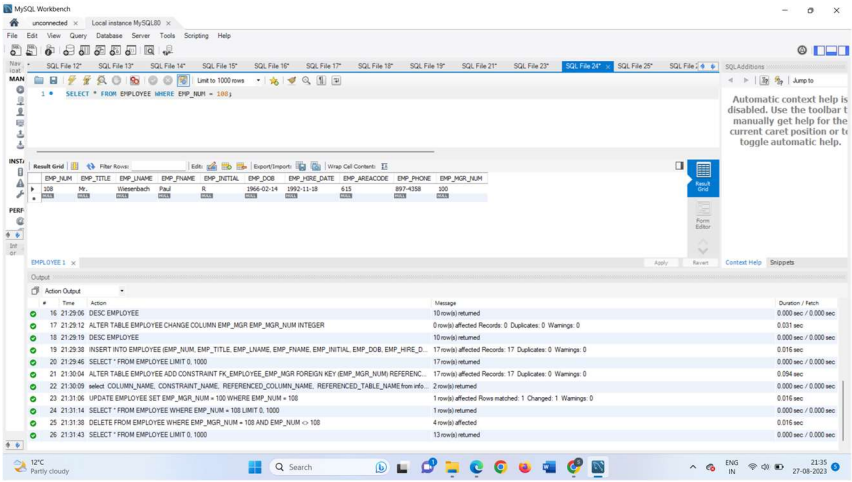
SQL Queries Demonstration

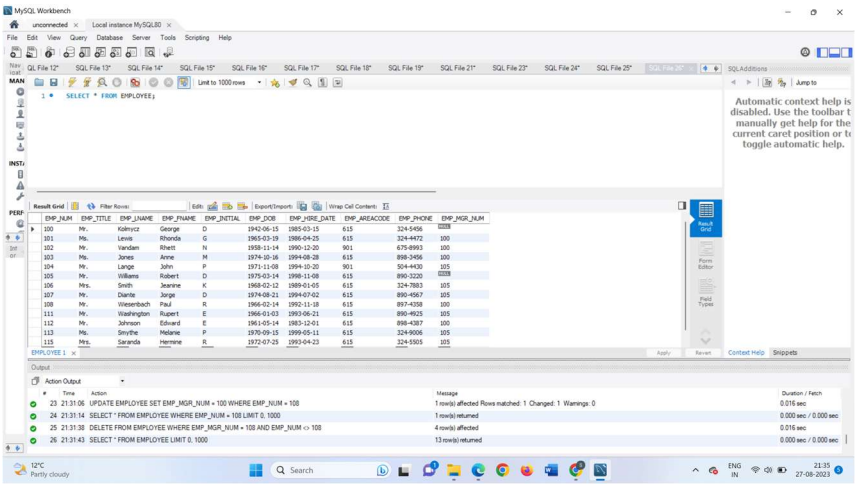
```
-- INNER JOIN with ORDER BY
SELECT M.FirstName, M.LastName, L.Title
FROM Member AS M
INNER JOIN LoanTransaction AS LT ON M.MemberID = LT.MemberID
INNER JOIN LibraryItem AS L ON LT.ItemID = L.ItemID
ORDER BY M.LastName;

-- WHERE with IN
SELECT ProgramName, ProgramDate, Location
FROM LibraryProgram
WHERE BranchID IN (SELECT BranchID FROM LibraryBranch WHERE BranchName = 'Main Library');

-- DATE function example
SELECT FirstName, LastName, DateOfBirth,
YEAR(CURRENT_DATE) - YEAR(DateOfBirth) -
CASE WHEN DATE_FORMAT(CURRENT_DATE, '%m-%d') > DATE_FORMAT(DateOfBirth, '%m-%d') THEN 0 ELSE
1 END AS Age
FROM Member;

-- View creation
CREATE VIEW BorrowedItems AS
SELECT M.FirstName, M.LastName, L.Title, LT.LoanDate, LT.ReturnDate
FROM Member AS M
INNER JOIN LoanTransaction AS LT ON M.MemberID = LT.MemberID
INNER JOIN LibraryItem AS L ON LT.ItemID = L.ItemID;
```

Some Outputs :




MySQL Workbench
select
COLUMN_NAME,
CONSTRAINT_NAME,
REFERENCED_COLUMN_NAME,
REFERENCED_TABLE_NAME from information_schema.KEY_COLUMN_USAGE where TABLE_NAME
= 'EMPLOYEE';
COLUMN_NAME CONSTRAINT_NAME REFERENCED_COLUMN_NAME REFERENCED_TABLE_NAME
EMP_NUM PRIMARY
EMP_MGR_NUM FK_EMPLOYEE_EMP_MGR EMP_NUM employee
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.


MySQL Workbench
SELECT * FROM EMPLOYEE WHERE EMP_NUM = 108;
EMP_NUM EMP_TITLE EMP_LNAME EMP_FNAME EMP_INITIAL EMP_DOB EMP_HIRE_DATE EMP_AREACODE EMP_PHONE EMP_MGR_NUM
108 Mr. Wiesenbach Paul R 1966-02-14 1992-11-18 615 897-4358 100
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.


MySQL Workbench
SELECT * FROM EMPLOYEE;
EMP_NUM EMP_TITLE EMP_LNAME EMP_FNAME EMP_INITIAL EMP_DOB EMP_HIRE_DATE EMP_AREACODE EMP_PHONE EMP_MGR_NUM
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.