

ITVEDANT EDUCATION

HYDERABAD Branch.



**Name:** RAKESH KUDACHI.

**COURSE:** MASTER IN FULL STACK DEVELOPER WITH JAVA.

**BATCH:** 20063.

**PROJECT NAME:** Library Management System using SQL.

**Title: Library Management System Project**

# Introduction:

This project demonstrates the implementation of a Library Management System using SQL. It includes creating and managing tables, performing CRUD operations, and executing advanced SQL queries. The goal is to showcase skills in database design, manipulation, and querying.

## **Objectives:**

1. **Set up the Library Management System Database**: Create and populate the database with tables for branches, employees, members, books, issued status, and return status.
2. **CRUD Operations**: Perform Create, Read, Update, and Delete operations on the data.
3. **CTAS (Create Table As Select)**: Utilize CTAS to create new tables based on query results.
4. **Advanced SQL Queries**: Develop complex queries to analyze and retrieve specific data.

## **Project Structure:**

### **1. Database Setup**

* **Database Creation**: Created a database named library\_db.
* **Table Creation**: Created tables for branches, employees, members, books, issued status, and return status. Each table includes relevant columns and relationships.

**CREATE DATABASE library\_db;**

DROP TABLE IF EXISTS branch;

**CREATE TABLE branch**

(

branch\_id VARCHAR(10) PRIMARY KEY,

manager\_id VARCHAR(10),

branch\_address VARCHAR(30),

contact\_no VARCHAR(15)

);

**-- Create table "Employee"**

DROP TABLE IF EXISTS employees;

CREATE TABLE employees

(

emp\_id VARCHAR(10) PRIMARY KEY,

emp\_name VARCHAR(30),

position VARCHAR(30),

salary DECIMAL(10,2),

branch\_id VARCHAR(10),

FOREIGN KEY (branch\_id) REFERENCES branch(branch\_id)

);

**-- Create table "Members"**

DROP TABLE IF EXISTS members;

CREATE TABLE members

(

member\_id VARCHAR(10) PRIMARY KEY,

member\_name VARCHAR(30),

member\_address VARCHAR(30),

reg\_date DATE

);

**-- Create table "Books"**

DROP TABLE IF EXISTS books;

CREATE TABLE books

(

isbn VARCHAR(50) PRIMARY KEY,

book\_title VARCHAR(80),

category VARCHAR(30),

rental\_price DECIMAL(10,2),

status VARCHAR(10),

author VARCHAR(30),

publisher VARCHAR(30)

);

**-- Create table "IssueStatus"**

DROP TABLE IF EXISTS issued\_status;

CREATE TABLE issued\_status

(

issued\_id VARCHAR(10) PRIMARY KEY,

issued\_member\_id VARCHAR(30),

issued\_book\_name VARCHAR(80),

issued\_date DATE,

issued\_book\_isbn VARCHAR(50),

issued\_emp\_id VARCHAR(10),

FOREIGN KEY (issued\_member\_id) REFERENCES members(member\_id),

FOREIGN KEY (issued\_emp\_id) REFERENCES employees(emp\_id),

FOREIGN KEY (issued\_book\_isbn) REFERENCES books(isbn)

);

**-- Create table "ReturnStatus"**

DROP TABLE IF EXISTS return\_status;

CREATE TABLE return\_status

(

return\_id VARCHAR(10) PRIMARY KEY,

issued\_id VARCHAR(30),

return\_book\_name VARCHAR(80),

return\_date DATE,

return\_book\_isbn VARCHAR(50),

FOREIGN KEY (return\_book\_isbn) REFERENCES books(isbn)

);

**-- Insert the values in the table --**

-- insert values into branch table --

insert into branch (branch\_id,manager\_id,branch\_address,contact\_no)

values ('B001','E109','Bengaluru','9099988676'),

('B002','E109','Belagavi','9099988677'),

('B003','E109','Pune','9099988678'),

('B004','E110','Hyderabad','9099988679'),

('B005','E110','Mumbai','9099988680');

**-- insert values into members table --**

INSERT INTO members(member\_id, member\_name, member\_address, reg\_date)

VALUES

('C101', 'Alice Johnson', '123 Main St', '2021-05-15'),

('C102', 'Bob Smith', '456 Elm St', '2021-06-20'),

('C103', 'Carol Davis', '789 Oak St', '2021-07-10'),

('C104', 'Dave Wilson', '567 Pine St', '2021-08-05'),

('C105', 'Eve Brown', '890 Maple St', '2021-09-25'),

('C106', 'Frank Thomas', '234 Cedar St', '2021-10-15'),

('C107', 'Grace Taylor', '345 Walnut St', '2021-11-20'),

('C108', 'Henry Anderson', '456 Birch St', '2021-12-10'),

('C109', 'Ivy Martinez', '567 Oak St', '2022-01-05'),

('C110', 'Jack Wilson', '678 Pine St', '2022-02-25'),

('C111', 'Sam', '133 Pine St', '2024-06-01'),

('C112', 'John', '143 Main St', '2024-05-01');

**-- insert values into employees table --**

insert into employees (emp\_id,emp\_name,position,salary,branch\_id)

values ("E101","John Doe","Clerk","60000.00","B001"),

("E102","Jane Smith","Clerk","45000.00","B002"),

("E103","Mike Johnson","Librarian","55000.00","B001"),

("E104","Emily Davis","Assistant","40000.00","B001"),

("E105","Sarah Brown","Assistant","42000.00","B002"),

("E106","Michelle Ramirez","Assistant","43000.00","B003"),

("E107","Michael Thompson","Clerk","62000.00","B005"),

("E108","Jessica Taylor","Clerk","46000.00","B004"),

("E109","Daniel Anderson","Manager","57000.00","B003"),

("E110","Laura Martinez","Manager","41000.00","B005"),

("E111","Christopher Lee","Assistant","65000.00","B005");

**-- insert values into a books table --**

insert into books(isbn,book\_title,category,rental\_price,status,author,publisher)

values ("978-0-553-29698-2","The Catcher in the Rye","Classic","7.00","yes","J.D. Salinger","Little, Brown and Company"),

("978-0-330-25864-8","Animal Farm","Classic","5.50","yes","George Orwell","Penguin Books"),

("978-0-14-118776-1","One Hundred Years of Solitude","Literary Fiction","6.50","yes","Gabriel Garcia Marquez","Penguin Books"),

("978-0-525-47535-5","The Great Gatsby","Classic","8.00","yes","F. Scott Fitzgerald","Scribner"),

("978-0-141-44171-6","Jane Eyre","Classic","4.00","yes","Charlotte Bronte","Penguin Classics"),

("978-0-307-37840-1","The Alchemist","Fiction","2.50","yes","Paulo Coelho","HarperOne"),

("978-0-679-76489-8","Harry Potter and the Sorcerers Stone","Fantasy","7.00","yes","J.K. Rowling","Scholastic"),

("978-0-7432-4722-4","The Da Vinci Code","Mystery","8.00","yes","Dan Brown","Doubleday"),

("978-0-09-957807-9","A Game of Thrones","Fantasy","7.50","yes","George R.R. Martin","Bantam"),

("978-0-393-05081-8","A Peoples History of the United States","History","9.00","yes","Howard Zinn","Harper Perennial"),

("978-0-19-280551-1","The Guns of August","History","7.00","yes","Barbara W. Tuchman","Oxford University Press"),

("978-0-307-58837-1","Sapiens: A Brief History of Humankind","History","8.00","no","Yuval Noah Harari","Harper Perennial"),

("978-0-375-41398-8","The Diary of a Young Girl","History","6.50","no","Anne Frank","Bantam"),

("978-0-14-044930-3","The Histories","History","5.50","yes","Herodotus","Penguin Classics"),

("978-0-393-91257-8","Guns, Germs, and Steel: The Fates of Human Societies","History","7.00","yes","Jared Diamond","W. W. Norton & Company"),

("978-0-7432-7357-1","1491: New Revelations of the Americas Before Columbus","History","6.50","no","Charles C. Mann","Vintage Books"),

("978-0-679-64115-3","1984","Dystopian","6.50","yes","George Orwell","Penguin Books"),

("978-0-14-143951-8","Pride and Prejudice","Classic","5.00","yes","Jane Austen","Penguin Classics"),

("978-0-452-28240-7","Brave New World","Dystopian","6.50","yes","Aldous Huxley","Harper Perennial"),

("978-0-670-81302-4","The Road","Dystopian","7.00","yes","Cormac McCarthy","Knopf"),

("978-0-385-33312-0","The Shining","Horror","6.00","yes","Stephen King","Doubleday"),

("978-0-451-52993-5","Fahrenheit 451","Dystopian","5.50","yes","Ray Bradbury","Ballantine Books"),

("978-0-345-39180-3","Dune","Science Fiction","8.50","yes","Frank Herbert","Ace"),

("978-0-375-50167-0","The Road","Dystopian","7.00","yes","Cormac McCarthy","Vintage"),

("978-0-06-025492-6","Where the Wild Things Are","Children","3.50","yes","Maurice Sendak","HarperCollins"),

("978-0-06-112241-5","The Kite Runner","Fiction","5.50","yes","Khaled Hosseini","Riverhead Books"),

("978-0-06-440055-8","Charlotte's Web","Children","4.00","yes","E.B. White","Harper & Row"),

("978-0-679-77644-3","Beloved","Fiction","6.50","yes","Toni Morrison","Knopf"),

("978-0-14-027526-3","A Tale of Two Cities","Classic","4.50","yes","Charles Dickens","Penguin Books"),

("978-0-7434-7679-3","The Stand","Horror","7.00","yes","Stephen King","Doubleday"),

("978-0-451-52994-2","Moby Dick","Classic","6.50","yes","Herman Melville","Penguin Books"),

("978-0-06-112008-4","To Kill a Mockingbird","Classic","5.00","yes","Harper Lee","J.B. Lippincott & Co."),

("978-0-553-57340-1","1984","Dystopian","6.50","yes","George Orwell","Penguin Books"),

("978-0-7432-4722-5","Angels & Demons","Mystery","7.50","yes","Dan Brown","Doubleday"),

("978-0-7432-7356-4","The Hobbit","Fantasy","7.00","yes","J.R.R. Tolkien","Houghton Mifflin Harcourt");

**-- insert values into issued\_status table --**

INSERT INTO issued\_status(issued\_id, issued\_member\_id, issued\_book\_name, issued\_date, issued\_book\_isbn, issued\_emp\_id)

VALUES

('IS106', 'C106', 'Animal Farm', '2024-03-10', '978-0-330-25864-8', 'E104'),

('IS107', 'C107', 'One Hundred Years of Solitude', '2024-03-11', '978-0-14-118776-1', 'E104'),

('IS108', 'C108', 'The Great Gatsby', '2024-03-12', '978-0-525-47535-5', 'E104'),

('IS109', 'C109', 'Jane Eyre', '2024-03-13', '978-0-141-44171-6', 'E105'),

('IS110', 'C110', 'The Alchemist', '2024-03-14', '978-0-307-37840-1', 'E105'),

('IS111', 'C109', 'Harry Potter and the Sorcerers Stone', '2024-03-15', '978-0-679-76489-8', 'E105'),

('IS112', 'C109', 'A Game of Thrones', '2024-03-16', '978-0-09-957807-9', 'E106'),

('IS113', 'C109', 'A Peoples History of the United States', '2024-03-17', '978-0-393-05081-8', 'E106'),

('IS114', 'C109', 'The Guns of August', '2024-03-18', '978-0-19-280551-1', 'E106'),

('IS115', 'C109', 'The Histories', '2024-03-19', '978-0-14-044930-3', 'E107'),

('IS116', 'C110', 'Guns, Germs, and Steel: The Fates of Human Societies', '2024-03-20', '978-0-393-91257-8', 'E107'),

('IS117', 'C110', '1984', '2024-03-21', '978-0-679-64115-3', 'E107'),

('IS118', 'C101', 'Pride and Prejudice', '2024-03-22', '978-0-14-143951-8', 'E108'),

('IS119', 'C110', 'Brave New World', '2024-03-23', '978-0-452-28240-7', 'E108'),

('IS120', 'C110', 'The Road', '2024-03-24', '978-0-670-81302-4', 'E108'),

('IS121', 'C102', 'The Shining', '2024-03-25', '978-0-385-33312-0', 'E109'),

('IS122', 'C102', 'Fahrenheit 451', '2024-03-26', '978-0-451-52993-5', 'E109'),

('IS123', 'C103', 'Dune', '2024-03-27', '978-0-345-39180-3', 'E109'),

('IS124', 'C104', 'Where the Wild Things Are', '2024-03-28', '978-0-06-025492-6', 'E110'),

('IS125', 'C105', 'The Kite Runner', '2024-03-29', '978-0-06-112241-5', 'E110'),

('IS126', 'C105', 'Charlotte''s Web', '2024-03-30', '978-0-06-440055-8', 'E110'),

('IS127', 'C105', 'Beloved', '2024-03-31', '978-0-679-77644-3', 'E110'),

('IS128', 'C105', 'A Tale of Two Cities', '2024-04-01', '978-0-14-027526-3', 'E110'),

('IS129', 'C105', 'The Stand', '2024-04-02', '978-0-7434-7679-3', 'E110'),

('IS130', 'C106', 'Moby Dick', '2024-04-03', '978-0-451-52994-2', 'E101'),

('IS131', 'C106', 'To Kill a Mockingbird', '2024-04-04', '978-0-06-112008-4', 'E101'),

('IS132', 'C106', 'The Hobbit', '2024-04-05', '978-0-7432-7356-4', 'E106'),

('IS133', 'C107', 'Angels & Demons', '2024-04-06', '978-0-7432-4722-5', 'E106'),

('IS134', 'C107', 'The Diary of a Young Girl', '2024-04-07', '978-0-375-41398-8', 'E106'),

('IS135', 'C107', 'Sapiens: A Brief History of Humankind', '2024-04-08', '978-0-307-58837-1', 'E108'),

('IS136', 'C107', '1491: New Revelations of the Americas Before Columbus', '2024-04-09', '978-0-7432-7357-1', 'E102'),

('IS137', 'C107', 'The Catcher in the Rye', '2024-04-10', '978-0-553-29698-2', 'E103'),

('IS138', 'C108', 'The Great Gatsby', '2024-04-11', '978-0-525-47535-5', 'E104'),

('IS139', 'C109', 'Harry Potter and the Sorcerers Stone', '2024-04-12', '978-0-679-76489-8', 'E105'),

('IS140', 'C110', 'Animal Farm', '2024-04-13', '978-0-330-25864-8', 'E102');

**-- insert values in return\_status --**

INSERT INTO return\_status(return\_id, issued\_id, return\_date)

VALUES

('RS101', 'IS121', '2023-06-06'),

('RS102', 'IS122', '2023-06-07'),

('RS103', 'IS123', '2023-08-07'),

('RS104', 'IS106', '2024-05-01'),

('RS105', 'IS107', '2024-05-03'),

('RS106', 'IS108', '2024-05-05'),

('RS107', 'IS109', '2024-05-07'),

('RS108', 'IS110', '2024-05-09'),

('RS109', 'IS111', '2024-05-11'),

('RS110', 'IS112', '2024-05-13'),

('RS111', 'IS113', '2024-05-15'),

('RS112', 'IS114', '2024-05-17'),

('RS113', 'IS115', '2024-05-19'),

('RS114', 'IS116', '2024-05-21'),

('RS115', 'IS117', '2024-05-23'),

('RS116', 'IS118', '2024-05-25'),

('RS117', 'IS119', '2024-05-27'),

('RS118', 'IS120', '2024-05-29');

### **2. CRUD Operations**

* **Create**: Inserted sample records into the books table.
* **Read**: Retrieved and displayed data from various tables.
* **Update**: Updated records in the employees table.
* **Delete**: Removed records from the members table as needed.

**Task 1. Create a New Book Record** -- "('978-1-60129-456-2', 'To Kill a Mockingbird', 'Classic', 6.00, 'yes', 'Harper Lee', 'J.B. Lippincott & Co.')"

INSERT INTO books(isbn, book\_title, category, rental\_price, status, author, publisher)

VALUES('978-1-60129-456-2', 'To Kill a Mockingbird', 'Classic', 6.00, 'yes', 'Harper Lee', 'J.B. Lippincott & Co.');

SELECT \* FROM books;

**Task 2: Update an Existing Member's Address**

UPDATE members

SET member\_address = '125 Oak St'

WHERE member\_id = 'C103';

**Task 3: Delete a Record from the Issued Status Table** -- Objective: Delete the record with issued\_id = 'IS121' from the issued\_status table.

DELETE FROM issued\_status

WHERE issued\_id = 'IS121';

**Task 4: Retrieve All Books Issued by a Specific Employee** -- Objective: Select all books issued by the employee with emp\_id = 'E101'.

SELECT \* FROM issued\_status

WHERE issued\_emp\_id = 'E101'

**Task 5: List Members Who Have Issued More Than One Book** -- Objective: Use GROUP BY to find members who have issued more than one book.

SELECT

issued\_emp\_id,

COUNT(\*)

FROM issued\_status

GROUP BY 1

HAVING COUNT(\*) > 1

### **3. CTAS (Create Table As Select):**

**Task 6: Create Summary Tables**: Used CTAS to generate new tables based on query results - each book and total book\_issued\_cnt\*\*

CREATE TABLE book\_issued\_cnt AS

SELECT b.isbn, b.book\_title, COUNT(ist.issued\_id) AS issue\_count

FROM issued\_status as ist

JOIN books as b

ON ist.issued\_book\_isbn = b.isbn

GROUP BY b.isbn, b.book\_title;

### **4. Data Analysis & Findings:**

The following SQL queries were used to address specific questions:

**Task 7. Retrieve All Books in a Specific Category**

SELECT \* FROM books

WHERE category = 'Classic';

**Task 8: Find Total Rental Income by Category**

SELECT

b.category,

SUM(b.rental\_price),

COUNT(\*)

FROM

issued\_status as ist

JOIN

books as b

ON b.isbn = ist.issued\_book\_isbn

GROUP BY 1

**Task 9: List Members Who Registered in the Last 180 Days**

SELECT \* FROM members

WHERE reg\_date >= CURRENT\_DATE - INTERVAL '180 days';

**Task 10: List Employees with Their Branch Manager's Name and their branch details**

SELECT

e1.emp\_id,

e1.emp\_name,

e1.position,

e1.salary,

b.\*,

e2.emp\_name as manager

FROM employees as e1

JOIN

branch as b

ON e1.branch\_id = b.branch\_id

JOIN

employees as e2

ON e2.emp\_id = b.manager\_id

**Task 11.** **Create a Table of Books with Rental Price Above a Certain Threshold**

CREATE TABLE expensive\_books AS

SELECT \* FROM books

WHERE rental\_price > 7.00;

Task 12: **Retrieve the List of Books Not Yet Returned**

SELECT \* FROM issued\_status as ist

LEFT JOIN

return\_status as rs

ON rs.issued\_id = ist.issued\_id

WHERE rs.return\_id IS NULL;

## **Advanced SQL Operations:**

**Task 13: Identify Members with Overdue Books**  
Write a query to identify members who have overdue books (assume a 30-day return period). Display the member's\_id, member's name, book title, issue date, and days overdue.

SELECT

ist.issued\_member\_id,

m.member\_name,

bk.book\_title,

ist.issued\_date,

-- rs.return\_date,

CURRENT\_DATE - ist.issued\_date as over\_dues\_days

FROM issued\_status as ist

JOIN

members as m

ON m.member\_id = ist.issued\_member\_id

JOIN

books as bk

ON bk.isbn = ist.issued\_book\_isbn

LEFT JOIN

return\_status as rs

ON rs.issued\_id = ist.issued\_id

WHERE

rs.return\_date IS NULL

AND

(CURRENT\_DATE - ist.issued\_date) > 30

ORDER BY 1

**Task 14: Update Book Status on Return**  
Write a query to update the status of books in the books table to "Yes" when they are returned (based on entries in the return\_status table).

Delimiter $$

CREATE PROCEDURE add\_return\_records(p\_return\_id VARCHAR(10), p\_issued\_id VARCHAR(10), p\_book\_quality VARCHAR(10))

AS

DECLARE

v\_isbn VARCHAR(50);

v\_book\_name VARCHAR(80);

BEGIN

-- all your logic and code

-- inserting into returns based on users input

INSERT INTO return\_status(return\_id, issued\_id, return\_date, book\_quality)

VALUES(p\_return\_id, p\_issued\_id, CURRENT\_DATE, p\_book\_quality);

SELECT

issued\_book\_isbn,

issued\_book\_name

INTO

v\_isbn,

v\_book\_name

FROM issued\_status

WHERE issued\_id = p\_issued\_id;

UPDATE books

SET status = 'yes'

WHERE isbn = v\_isbn;

RAISE NOTICE 'Thank you for returning the book: %', v\_book\_name;

END ;

Delimiter $$

-- Testing FUNCTION add\_return\_records

issued\_id = IS135

ISBN = WHERE isbn = '978-0-307-58837-1'

SELECT \* FROM books

WHERE isbn = '978-0-307-58837-1';

SELECT \* FROM issued\_status

WHERE issued\_book\_isbn = '978-0-307-58837-1';

SELECT \* FROM return\_status

WHERE issued\_id = 'IS135';

-- calling function

CALL add\_return\_records('RS138', 'IS135', 'Good');

-- calling function

CALL add\_return\_records('RS148', 'IS140', 'Good');

**Task 15: Branch Performance Report**  
Create a query that generates a performance report for each branch, showing the number of books issued, the number of books returned, and the total revenue generated from book rentals.

CREATE TABLE branch\_reports

AS

SELECT

b.branch\_id,

b.manager\_id,

COUNT(ist.issued\_id) as number\_book\_issued,

COUNT(rs.return\_id) as number\_of\_book\_return,

SUM(bk.rental\_price) as total\_revenue

FROM issued\_status as ist

JOIN

employees as e

ON e.emp\_id = ist.issued\_emp\_id

JOIN

branch as b

ON e.branch\_id = b.branch\_id

LEFT JOIN

return\_status as rs

ON rs.issued\_id = ist.issued\_id

JOIN

books as bk

ON ist.issued\_book\_isbn = bk.isbn

GROUP BY 1, 2;

SELECT \* FROM branch\_reports;

**Task 16: CTAS: Create a Table of Active Members**  
Use the CREATE TABLE AS (CTAS) statement to create a new table active\_members containing members who have issued at least one book in the last 2 months.

CREATE TABLE active\_members

AS

SELECT \* FROM members

WHERE member\_id IN (SELECT

DISTINCT issued\_member\_id

FROM issued\_status

WHERE

issued\_date >= CURRENT\_DATE - INTERVAL '2 month'

)

;

SELECT \* FROM active\_members;

**Task 17: Find Employees with the Most Book Issues Processed**  
Write a query to find the top 3 employees who have processed the most book issues. Display the employee name, number of books processed, and their branch.

SELECT

e.emp\_name,

b.\*,

COUNT(ist.issued\_id) as no\_book\_issued

FROM issued\_status as ist

JOIN

employees as e

ON e.emp\_id = ist.issued\_emp\_id

JOIN

branch as b

ON e.branch\_id = b.branch\_id

GROUP BY 1, 2

**Task 18: Identify Members Issuing High-Risk Books**  
Write a query to identify members who have issued books more than twice with the status "damaged" in the books table. Display the member name, book title, and the number of times they've issued damaged books.

SELECT

m.member\_id,

m.member\_name,

ist.issued\_book\_name,

COUNT(\*)

FROM return\_status AS rs

JOIN issued\_status AS ist ON rs.issued\_id = ist.issued\_id

JOIN members AS m ON m.member\_id = ist.issued\_member\_id

WHERE rs.book\_quality = 'Damaged'

GROUP BY 1, 3

HAVING COUNT(\*) > 2;

## Reports:

-- **Database Schema**: Detailed table structures and relationships. -- **Data Analysis**: Insights into book categories, employee salaries, member registration trends, and issued books. -- **Summary Reports**: Aggregated data on high-demand books and employee performance.

## **Conclusion:**

This project demonstrates the application of SQL skills in creating and managing a library management system. It includes database setup, data manipulation, and advanced querying, providing a solid foundation -- for data management and analysis.