1)

CREATE TABLE PUBLISHER (

Name VARCHAR(100) PRIMARY KEY,

Address VARCHAR(255),

Phone VARCHAR(20)

);

CREATE TABLE BOOK (

Book\_id INT PRIMARY KEY,

Title VARCHAR(255),

Publisher\_Name VARCHAR(100),

Pub\_Year INT,

FOREIGN KEY (Publisher\_Name) REFERENCES PUBLISHER(Name)

);

CREATE TABLE BOOK\_AUTHORS (

Book\_id INT,

Author\_Name VARCHAR(100),

FOREIGN KEY (Book\_id) REFERENCES BOOK(Book\_id)

);

CREATE TABLE LIBRARY\_BRANCH (

Branch\_id INT PRIMARY KEY,

Branch\_Name VARCHAR(100),

Address VARCHAR(255)

);

CREATE TABLE BOOK\_COPIES (

Book\_id INT,

Branch\_id INT,

No\_of\_Copies INT,

FOREIGN KEY (Book\_id) REFERENCES BOOK(Book\_id),

FOREIGN KEY (Branch\_id) REFERENCES LIBRARY\_BRANCH(Branch\_id)

);

CREATE TABLE CARD (

Card\_No INT PRIMARY KEY

);

CREATE TABLE BOOK\_LENDING (

Book\_id INT,

Branch\_id INT,

Card\_No INT,

Date\_Out DATE,

Due\_Date DATE,

FOREIGN KEY (Book\_id) REFERENCES BOOK(Book\_id),

FOREIGN KEY (Branch\_id) REFERENCES LIBRARY\_BRANCH(Branch\_id),

FOREIGN KEY (Card\_No) REFERENCES CARD(Card\_No)

);

-- 2. Insert Sample Data (5 sample entries for each)

-- Publishers

INSERT INTO PUBLISHER VALUES

('Penguin', 'New York', '1234567890'),

('HarperCollins', 'London', '0987654321'),

('OUP', 'Oxford', '1111111111');

-- Books

INSERT INTO BOOK VALUES

(1, 'The Alchemist', 'HarperCollins', 1988),

(2, 'Animal Farm', 'Penguin', 1945),

(3, '1984', 'Penguin', 1949);

-- Book Authors

INSERT INTO BOOK\_AUTHORS VALUES

(1, 'Paulo Coelho'),

(2, 'George Orwell'),

(3, 'George Orwell');

-- Branches

INSERT INTO LIBRARY\_BRANCH VALUES

(101, 'Central Library', 'MG Road'),

(102, 'North Branch', 'Indiranagar');

-- Book Copies

INSERT INTO BOOK\_COPIES VALUES

(1, 101, 3),

(2, 101, 5),

(3, 102, 2);

-- Cards

INSERT INTO CARD VALUES

(201), (202), (203);

-- Lending records

INSERT INTO BOOK\_LENDING VALUES

(1, 101, 201, '2020-01-10', '2020-01-24'),

(2, 101, 201, '2020-03-15', '2020-03-30'),

(3, 102, 201, '2020-06-20', '2020-07-05'),

(1, 101, 202, '2021-05-10', '2021-05-20');

-- 3. a. Retrieve all book details with authors and copies

SELECT

B.Book\_id,

B.Title,

B.Publisher\_Name,

BA.Author\_Name,

BC.Branch\_id,

LB.Branch\_Name,

BC.No\_of\_Copies

FROM

BOOK B

LEFT JOIN BOOK\_AUTHORS BA ON B.Book\_id = BA.Book\_id

LEFT JOIN BOOK\_COPIES BC ON B.Book\_id = BC.Book\_id

LEFT JOIN LIBRARY\_BRANCH LB ON BC.Branch\_id = LB.Branch\_id;

-- 4. b. Borrowers who borrowed more than 2 books in 2020

SELECT

Card\_No,

COUNT(\*) AS Total\_Borrowed

FROM

BOOK\_LENDING

WHERE

YEAR(Date\_Out) = 2020

GROUP BY

Card\_No

HAVING

COUNT(\*) > 2;

-- 5. c. Delete a book and update other tables

-- Assume deleting Book\_id = 1

DELETE FROM BOOK\_LENDING WHERE Book\_id = 1;

DELETE FROM BOOK\_COPIES WHERE Book\_id = 1;

DELETE FROM BOOK\_AUTHORS WHERE Book\_id = 1;

DELETE FROM BOOK WHERE Book\_id = 1;

-- 6. d. Total number of books published by each publisher

SELECT

Publisher\_Name,

COUNT(\*) AS Total\_Books

FROM

BOOK

GROUP BY

Publisher\_Name;

-- 7. e. Create view for all books and number of copies available

CREATE VIEW Available\_Books AS

SELECT

B.Book\_id,

B.Title,

SUM(BC.No\_of\_Copies) AS Total\_Copies

FROM

BOOK B

JOIN BOOK\_COPIES BC ON B.Book\_id = BC.Book\_id

GROUP BY

B.Book\_id, B.Title;

-- View the result

SELECT \* FROM Available\_Books;

2) -----------------------------------------------------------------------------------------------------------------------------

CREATE TABLE EMPLOYEE (

SSN INT PRIMARY KEY,

Name VARCHAR(100),

Address VARCHAR(255),

Gender CHAR(1),

Salary DECIMAL(10,2),

SuperSSN INT,

DNo INT

);

-- Create DEPARTMENT table

CREATE TABLE DEPARTMENT (

DNo INT PRIMARY KEY,

DName VARCHAR(100),

MgrSSN INT,

MgrStartDate DATE

);

-- Create DLOCATION table

CREATE TABLE DLOCATION (

DNo INT,

DLoc VARCHAR(100),

FOREIGN KEY (DNo) REFERENCES DEPARTMENT(DNo)

);

-- Create PROJECT table

CREATE TABLE PROJECT (

PNo INT PRIMARY KEY,

PName VARCHAR(100),

PLocation VARCHAR(100),

DNo INT,

FOREIGN KEY (DNo) REFERENCES DEPARTMENT(DNo)

);

-- Create WORKS\_ON table

CREATE TABLE WORKS\_ON (

SSN INT,

PNo INT,

Hours DECIMAL(5,2),

FOREIGN KEY (SSN) REFERENCES EMPLOYEE(SSN),

FOREIGN KEY (PNo) REFERENCES PROJECT(PNo)

);

-- Departments

INSERT INTO DEPARTMENT VALUES

(1, 'Accounts', 1001, '2018-01-01'),

(5, 'Research', 1002, '2019-03-15');

-- Employees

INSERT INTO EMPLOYEE VALUES

(1001, 'Michael Scott', 'Scranton', 'M', 70000, NULL, 1),

(1002, 'Jim Halpert', 'Scranton', 'M', 50000, 1001, 5),

(1003, 'Pam Beesly', 'Scranton', 'F', 45000, 1002, 5),

(1004, 'Dwight Schrute', 'Scranton', 'M', 60000, 1001, 5);

-- Projects

INSERT INTO PROJECT VALUES

(101, 'Inventory', 'Scranton', 1),

(102, 'loT', 'Stamford', 5),

(103, 'Security', 'Scranton', 5);

-- Work assignments

INSERT INTO WORKS\_ON VALUES

(1002, 102, 20),

(1003, 102, 15),

(1004, 102, 30),

(1003, 103, 10),

(1004, 103, 15);

--a

CREATE OR REPLACE TRIGGER trg\_uppercase\_name

BEFORE INSERT OR UPDATE ON EMPLOYEE

FOR EACH ROW

BEGIN

:NEW.Name := UPPER(:NEW.Name);

END;

/

-- As Worker or Manager

SELECT DISTINCT P.PNo

FROM PROJECT P

JOIN DEPARTMENT D ON P.DNo = D.DNo

LEFT JOIN EMPLOYEE MGR ON D.MgrSSN = MGR.SSN

LEFT JOIN WORKS\_ON W ON P.PNo = W.PNo

LEFT JOIN EMPLOYEE E ON W.SSN = E.SSN

WHERE

MGR.Name LIKE '%SCOTT%' OR

E.Name LIKE '%SCOTT%';

SELECT

E.SSN,

E.Name,

E.Salary AS Original\_Salary,

ROUND(E.Salary \* 1.10, 2) AS New\_Salary

FROM

EMPLOYEE E

WHERE

E.SSN IN (

SELECT W.SSN

FROM WORKS\_ON W

JOIN PROJECT P ON W.PNo = P.PNo

WHERE P.PName = 'loT'

);

SELECT

SUM(E.Salary) AS Total\_Salary,

MAX(E.Salary) AS Max\_Salary,

MIN(E.Salary) AS Min\_Salary,

AVG(E.Salary) AS Avg\_Salary

FROM

EMPLOYEE E

JOIN DEPARTMENT D ON E.DNo = D.DNo

WHERE

D.DName = 'Accounts';

SELECT E.Name

FROM EMPLOYEE E

WHERE NOT EXISTS (

SELECT P.PNo

FROM PROJECT P

WHERE P.DNo = 5

AND NOT EXISTS (

SELECT \* FROM WORKS\_ON W

WHERE W.PNo = P.PNo AND W.SSN = E.SSN

)

);

3)------------------------------------------------------------------------------------------

-- CUSTOMER Table

CREATE TABLE CUSTOMER (

CUSTOMER\_ID NUMBER(10) PRIMARY KEY,

NAME VARCHAR(20),

TRANSACTION\_DATE DATE

);

-- ACCOUNT Table

CREATE TABLE ACCOUNT (

ACCOUNT\_NO NUMBER(10) PRIMARY KEY,

TYPE VARCHAR(20),

BALANCE NUMBER(10),

TRANSACTION\_DATE DATE

);

-- ADDRESS Table

CREATE TABLE ADDRESS (

CUSTOMER\_ID NUMBER(10),

STREET VARCHAR(20),

CITY VARCHAR(20),

STATE VARCHAR(20),

PRIMARY KEY (CUSTOMER\_ID, STREET),

FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID) ON DELETE CASCADE

);

-- CUSTOMER\_ACCOUNT Table (many-to-many relationship)

CREATE TABLE CUSTOMER\_ACCOUNT (

CUSTOMER\_ID NUMBER(10),

ACCOUNT\_NO NUMBER(10),

PRIMARY KEY (CUSTOMER\_ID, ACCOUNT\_NO),

FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID) ON DELETE CASCADE,

FOREIGN KEY (ACCOUNT\_NO) REFERENCES ACCOUNT(ACCOUNT\_NO) ON DELETE CASCADE

);

-- Insert into CUSTOMER

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(1, 'Alice', TO\_DATE('2024-01-01', 'YYYY-MM-DD'));

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(2, 'Bob', TO\_DATE('2024-01-02', 'YYYY-MM-DD'));

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(3, 'Charlie', TO\_DATE('2024-01-03', 'YYYY-MM-DD'));

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(4, 'David', TO\_DATE('2024-01-04', 'YYYY-MM-DD'));

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(5, 'Eva', TO\_DATE('2024-01-05', 'YYYY-MM-DD'));

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, TRANSACTION\_DATE) VALUES

(6, 'Frank', TO\_DATE('2024-01-06', 'YYYY-MM-DD'));

-- Insert into ACCOUNT

INSERT INTO ACCOUNT (ACCOUNT\_NO, TYPE, BALANCE, TRANSACTION\_DATE) VALUES

(101, 'Savings', 5000, TO\_DATE('2024-01-10', 'YYYY-MM-DD'));

INSERT INTO ACCOUNT (ACCOUNT\_NO, TYPE, BALANCE, TRANSACTION\_DATE) VALUES

(102, 'Current', 15000, TO\_DATE('2024-01-11', 'YYYY-MM-DD'));

INSERT INTO ACCOUNT (ACCOUNT\_NO, TYPE, BALANCE, TRANSACTION\_DATE) VALUES

(103, 'Savings', 8000, TO\_DATE('2024-01-12', 'YYYY-MM-DD'));

-- Insert into ADDRESS

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(1, '1st Street', 'Mumbai', 'MH');

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(2, '2nd Street', 'Delhi', 'DL');

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(3, '3rd Street', 'Chennai', 'TN');

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(4, '4th Street', 'Kolkata', 'WB');

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(5, '5th Street', 'Bangalore', 'KA');

INSERT INTO ADDRESS (CUSTOMER\_ID, STREET, CITY, STATE) VALUES

(6, '6th Street', 'Hyderabad', 'TS');

-- Insert into CUSTOMER\_ACCOUNT (linking 4 customers to account 101)

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (1, 101);

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (2, 101);

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (3, 101);

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (4, 101);

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (5, 102);

INSERT INTO CUSTOMER\_ACCOUNT (CUSTOMER\_ID, ACCOUNT\_NO) VALUES (6, 103);

--b

UPDATE ACCOUNT

SET BALANCE = BALANCE \* 1.05

WHERE BALANCE < 10000;

--c

SELECT CA.ACCOUNT\_NO, COUNT(CA.CUSTOMER\_ID) AS NUM\_CUSTOMERS

FROM CUSTOMER\_ACCOUNT CA

GROUP BY CA.ACCOUNT\_NO

HAVING COUNT(CA.CUSTOMER\_ID) > 3;

--d

SELECT C.CUSTOMER\_ID, SUM(0.5 \* A.BALANCE) AS TOTAL\_INTEREST

FROM CUSTOMER C

JOIN CUSTOMER\_ACCOUNT CA ON C.CUSTOMER\_ID = CA.CUSTOMER\_ID

JOIN ACCOUNT A ON A.ACCOUNT\_NO = CA.ACCOUNT\_NO

GROUP BY C.CUSTOMER\_ID;

--E

SELECT C.CUSTOMER\_ID, C.NAME

FROM CUSTOMER C

JOIN CUSTOMER\_ACCOUNT CA ON C.CUSTOMER\_ID = CA.CUSTOMER\_ID

JOIN ACCOUNT A ON A.ACCOUNT\_NO = CA.ACCOUNT\_NO

WHERE A.TRANSACTION\_DATE IS NULL;

4)---------------------------------------------------------------------------------

-- Subject (Module) Table

CREATE TABLE SUBJECT (

SUBJECT\_CODE VARCHAR(10) PRIMARY KEY,

TITLE VARCHAR(100),

CREDIT NUMBER,

MODULE\_LEADER\_ID NUMBER,

DEPT\_ID NUMBER,

PREREQUISITE\_CODE VARCHAR(10),

FOREIGN KEY (MODULE\_LEADER\_ID) REFERENCES TEACHER(TEACHER\_ID),

FOREIGN KEY (DEPT\_ID) REFERENCES DEPARTMENT(DEPT\_ID),

FOREIGN KEY (PREREQUISITE\_CODE) REFERENCES SUBJECT(SUBJECT\_CODE)

);

-- Subject\_Teacher mapping (for teaching staff)

CREATE TABLE SUBJECT\_TEACHER (

SUBJECT\_CODE VARCHAR(10),

TEACHER\_ID NUMBER,

PRIMARY KEY (SUBJECT\_CODE, TEACHER\_ID),

FOREIGN KEY (SUBJECT\_CODE) REFERENCES SUBJECT(SUBJECT\_CODE),

FOREIGN KEY (TEACHER\_ID) REFERENCES TEACHER(TEACHER\_ID)

);

-- Student Table

CREATE TABLE STUDENT (

SERIAL\_NO NUMBER PRIMARY KEY,

NAME VARCHAR(50),

ADDRESS VARCHAR(100)

);

-- Student\_Subject Table (for performance)

CREATE TABLE STUDENT\_SUBJECT (

SERIAL\_NO NUMBER,

SUBJECT\_CODE VARCHAR(10),

MARKS NUMBER,

PRIMARY KEY (SERIAL\_NO, SUBJECT\_CODE),

FOREIGN KEY (SERIAL\_NO) REFERENCES STUDENT(SERIAL\_NO),

FOREIGN KEY (SUBJECT\_CODE) REFERENCES SUBJECT(SUBJECT\_CODE)

);

-- Departments

INSERT INTO DEPARTMENT VALUES (1, 'CSE');

INSERT INTO DEPARTMENT VALUES (2, 'ISE');

-- Teachers

INSERT INTO TEACHER VALUES (101, 'Dr. Smith', 1);

INSERT INTO TEACHER VALUES (102, 'Dr. Ramesh', 1);

INSERT INTO TEACHER VALUES (103, 'Prof. Anita', 2);

INSERT INTO TEACHER VALUES (104, 'Prof. Raj', 2);

-- Subjects

INSERT INTO SUBJECT VALUES ('CS101', 'Database Management System', 4, 101, 1, NULL);

INSERT INTO SUBJECT VALUES ('CS102', 'Operating Systems', 3, 102, 1, 'CS101');

INSERT INTO SUBJECT VALUES ('IS201', 'Data Structures', 3, 103, 2, NULL);

-- Subject Teachers

INSERT INTO SUBJECT\_TEACHER VALUES ('CS101', 101);

INSERT INTO SUBJECT\_TEACHER VALUES ('CS101', 102);

INSERT INTO SUBJECT\_TEACHER VALUES ('CS102', 102);

INSERT INTO SUBJECT\_TEACHER VALUES ('IS201', 103);

INSERT INTO SUBJECT\_TEACHER VALUES ('IS201', 104);

-- Students

INSERT INTO STUDENT VALUES (1, 'Alice', 'Bangalore');

INSERT INTO STUDENT VALUES (2, 'Bob', 'Mysore');

INSERT INTO STUDENT VALUES (3, 'Charlie', 'Delhi');

-- Student Marks

INSERT INTO STUDENT\_SUBJECT VALUES (1, 'CS101', 85);

INSERT INTO STUDENT\_SUBJECT VALUES (1, 'CS102', 65);

INSERT INTO STUDENT\_SUBJECT VALUES (2, 'CS101', 72);

INSERT INTO STUDENT\_SUBJECT VALUES (2, 'IS201', 50);

INSERT INTO STUDENT\_SUBJECT VALUES (3, 'CS102', 35);

SELECT NAME

FROM TEACHER

WHERE TEACHER\_ID NOT IN (

SELECT MODULE\_LEADER\_ID FROM SUBJECT

);

SELECT D.DEPT\_NAME

FROM DEPARTMENT D

JOIN SUBJECT S ON S.DEPT\_ID = D.DEPT\_ID

WHERE S.TITLE = 'Database Management System';

SELECT T.NAME, COUNT(ST.SUBJECT\_CODE) AS SUBJECTS\_TAUGHT

FROM TEACHER T

JOIN SUBJECT\_TEACHER ST ON T.TEACHER\_ID = ST.TEACHER\_ID

GROUP BY T.NAME;

SELECT

S.SERIAL\_NO,

S.NAME,

SS.SUBJECT\_CODE,

SS.MARKS,

CASE

WHEN SS.MARKS BETWEEN 70 AND 100 THEN 'Outstanding'

WHEN SS.MARKS BETWEEN 40 AND 69 THEN 'Average'

ELSE 'Weak'

END AS CATEGORY

FROM STUDENT S

JOIN STUDENT\_SUBJECT SS ON S.SERIAL\_NO = SS.SERIAL\_NO;