Library Management System

DBMS Group Project -1

Table of contents

01

Problem Statement

You can enter a subtitle here if you need it

DATABASE NAME: **Library Management System database:**

The Issue and Return of books at a library is the most difficult task to manage. An automated system with a powerful database will make things easier for an institution as well as for users. Students will be able to return books and will be able to save themselves from the applied fines.

02

Assumptions

You can enter a subtitle here if you need it

Assumptions in our Data base

1. Each book has a unique ID number
2. Members of the library can search for books by title, author, category, and publication year
3. A book can have multiple authors
4. There may be more than one copy of a book owned by the library
5. Members can borrow books, and the system will store the date that they borrowed the book.
6. Library staff can see who has borrowed a particular book, who has checked out a book in the past, when members have joined, and the status of a member.
7. Members can reserve copies of the books to borrow later, if all of the library's copies are borrowed by other members
8. Fines can be imposed on members if books are not returned within 7 days of borrowing them.
9. Members can pay the fines that have been added to their account.

Rules of converting ER model to Relational Model

- Entity type is converted to a Relation table.
- 1:1 or 1: N relationship type is converted to foreign key.
- M: N relationship type is converted to a relation with two foreign key.
- Simple attribute converted to an attribute.
- Value set converted to a domain.
- Key attribute converted to a primary key.

Awesome words

"Transforming information into inspiration, our library database management system fuels the flames of curiosity, empowering minds to explore, discover, and imagine boundless worlds within the pages of knowledge."

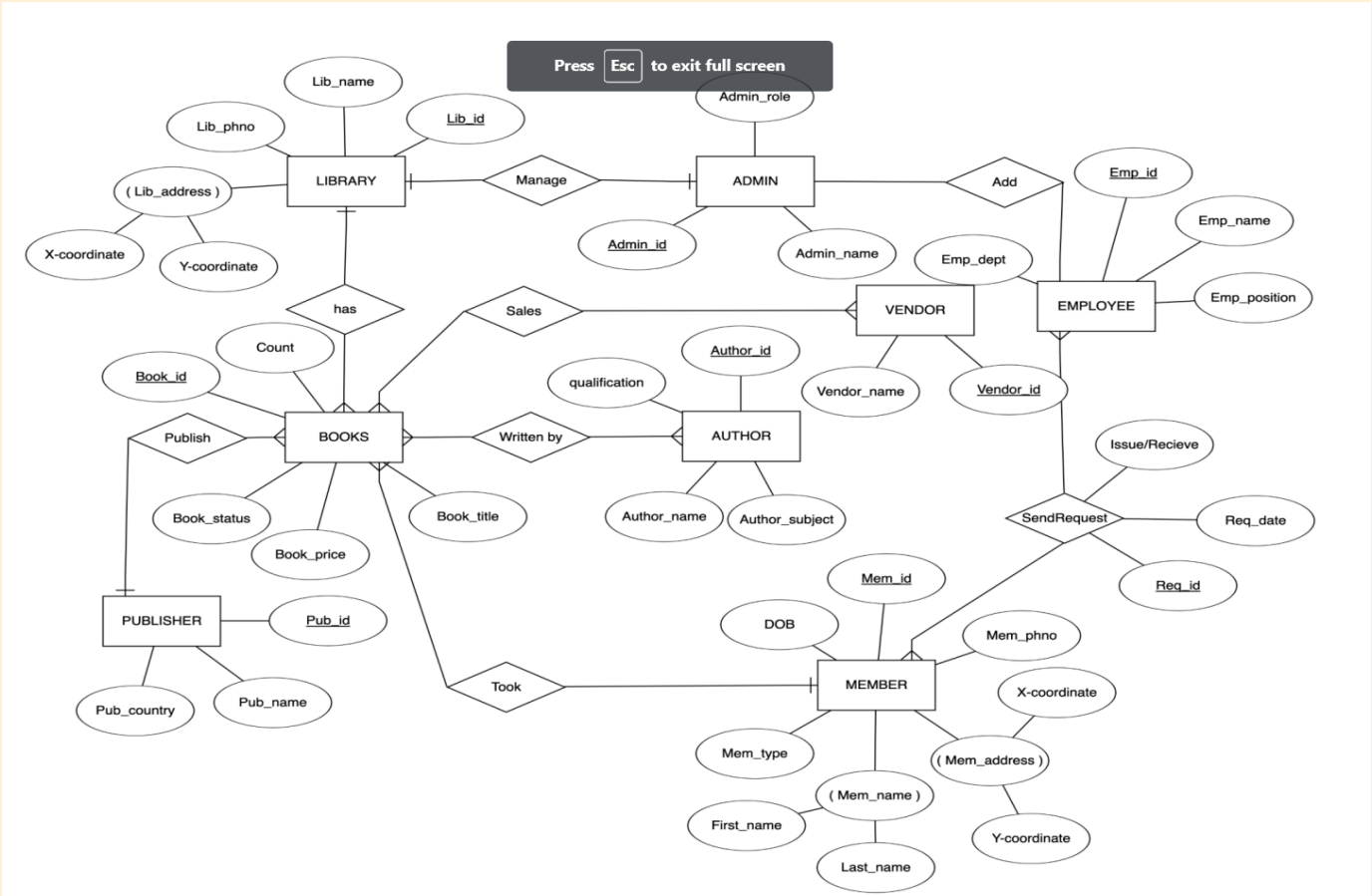
—Someone Famous

03

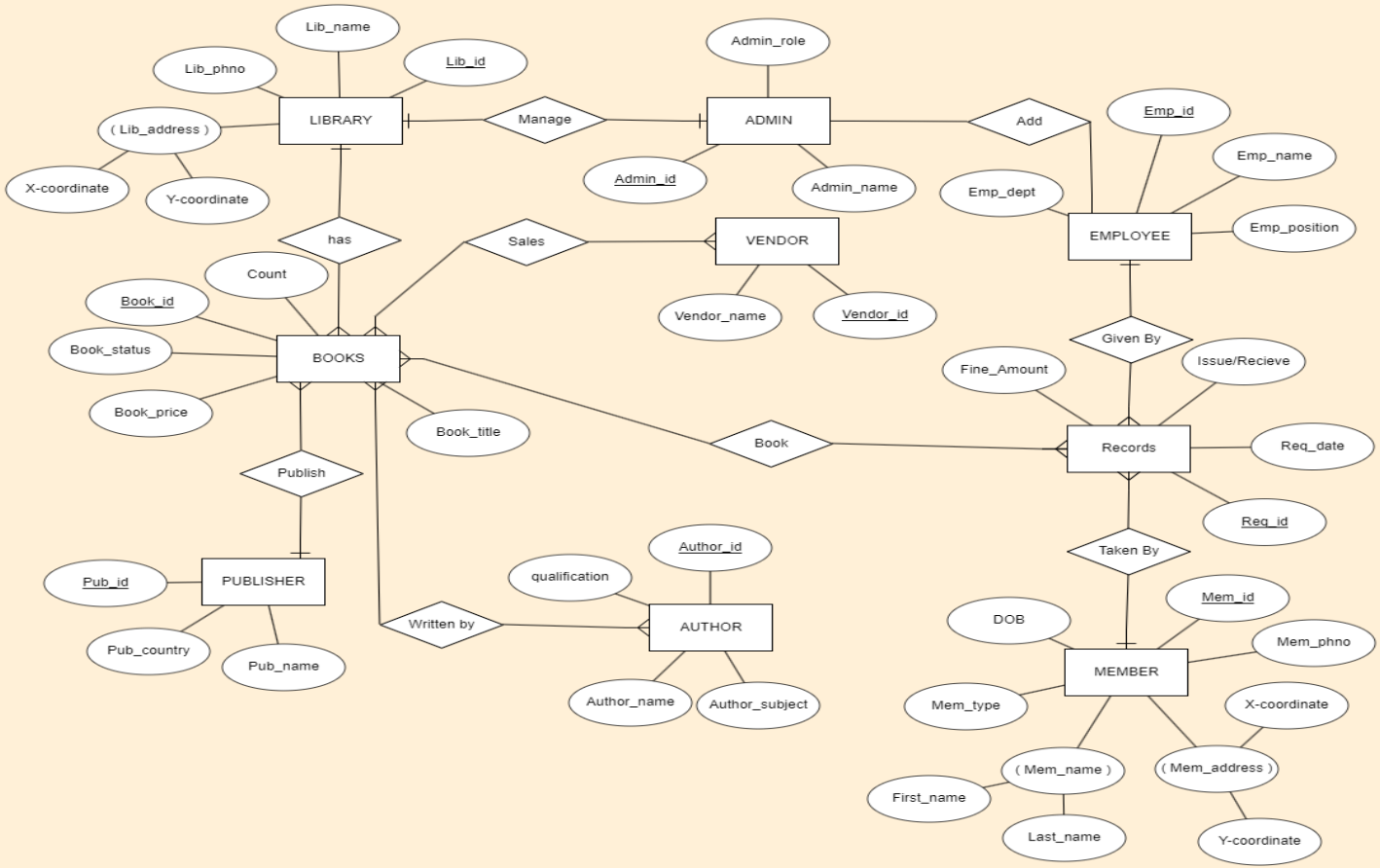
ER & Relational Model

You can enter a subtitle here if you need it

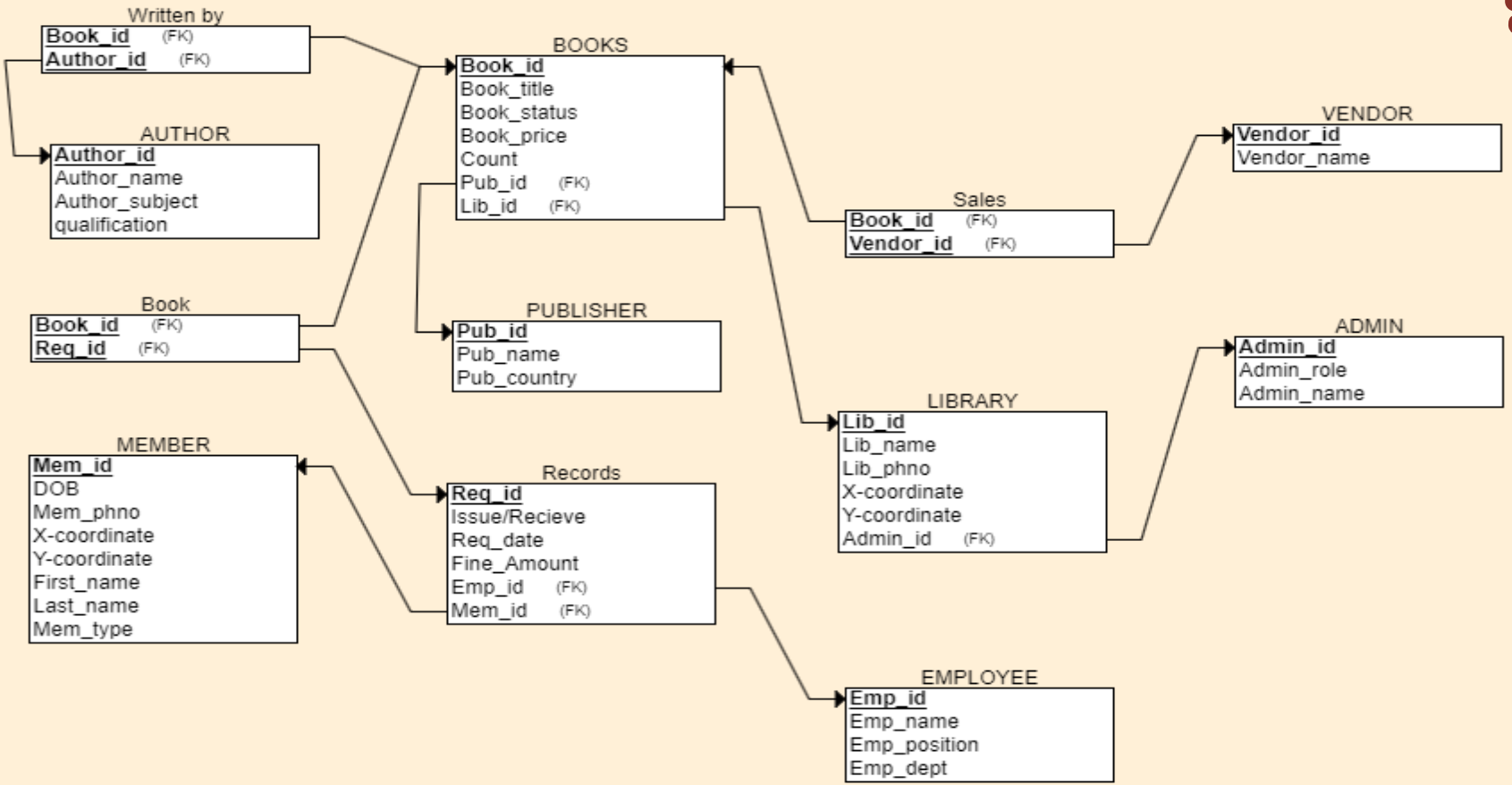
ER MODEL BEFORE NORMALIZATION


Press Esc to exit full screen
Lib_name
Lib_phno
Lib_id
LIBRARY
(Lib_address)
X-coordinate
Y-coordinate
Manage
Admin_role
ADMIN
Admin_id
Admin_name
Add
Emp_id
Emp_name
Emp_dept
EMPLOYEE
Emp_position
has
Sales
VENDOR
Vendor_name
Vendor_id
Count
Book_id
Author_id
qualification
Publish
BOOKS
Written by
AUTHOR
Book_status
Book_price
Book_title
Author_name
Author_subject
Issue/Recieve
SendRequest
Req_date
Req_id
PUBLISHER
Pub_id
Pub_country
Pub_name
Mem_id
DOB
Mem_phno
Took
MEMBER
X-coordinate
Mem_type
(Mem_address)
(Mem_name)
First_name
Last_name
Y-coordinate

ER MODEL


Lib_name
Lib_phno
Lib_id
LIBRARY
(Lib_address)
X-coordinate
Y-coordinate
Manage
Admin_role
ADMIN
Admin_id
Admin_name
Add
Emp_id
Emp_name
Emp_dept
EMPLOYEE
Emp_position
has
Sales
VENDOR
Vendor_name
Vendor_id
Count
Book_id
Book_status
BOOKS
Book_price
Book_title
Publish
Book
Given By
Fine_Amount
Issue/Recieve
Records
Req_date
Req_id
Taken By
Mem_id
DOB
Mem_phno
MEMBER
Mem_type
X-coordinate
(Mem_name)
(Mem_address)
First_name
Last_name
Y-coordinate
Pub_id
PUBLISHER
Pub_country
Pub_name
Written by
qualification
Author_id
AUTHOR
Author_name
Author_subject

RELATIONAL MODEL


Written by
Book_id (FK)
Author_id (FK)
AUTHOR
Author_id
Author_name
Author_subject
qualification
Book
Book_id (FK)
Req_id (FK)
MEMBER
Mem_id
DOB
Mem_phno
X-coordinate
Y-coordinate
First_name
Last_name
Mem_type
BOOKS
Book_id
Book_title
Book_status
Book_price
Count
Pub_id (FK)
Lib_id (FK)
PUBLISHER
Pub_id
Pub_name
Pub_country
Records
Req_id
Issue/Recieve
Req_date
Fine_Amount
Emp_id (FK)
Mem_id (FK)
Sales
Book_id (FK)
Vendor_id (FK)
VENDOR
Vendor_id
Vendor_name
LIBRARY
Lib_id
Lib_name
Lib_phno
X-coordinate
Y-coordinate
Admin_id (FK)
ADMIN
Admin_id
Admin_role
Admin_name
EMPLOYEE
Emp_id
Emp_name
Emp_position
Emp_dept

04

Functional Dependencies

You can enter a subtitle here if you need it

1)LIBRARY table:

Functional Dependencies:
Lib_id -> Lib_phno, Lib_name, Lib_address_x, Lib_address_y

2)BOOKS table:

Functional Dependencies:
Book_id -> Count, Book_status, Book_price, Book_title, Lib_id, Pub_id

3)ADMIN table:

Functional Dependencies:
Admin_id -> Admin_name, Admin_role

4)EMPLOYEE table:

Functional Dependencies:
Emp_id -> Emp_name, Emp_position, Emp_dept

5)VENDOR table:

Functional Dependencies:
Vendor_id -> Vendor_name

6)RECORDS table:

Functional Dependencies:
Req_id -> Received_date, issued_date, fine, Book_id, Mem_id, Emp_id

7)MEMBER table:

Functional Dependencies:
Mem_id -> Mem_phno, DOB, Mem_type, Mem_first_name, Mem_last_name, Mem_address_x, Mem_address_y

8)book table:

Functional Dependencies:
(Book_id, Req_id) -> No additional functional dependencies beyond the primary key.

9)AUTHOR table:

Functional Dependencies:
Author_id -> qualification, Author_name, Author_subject

10)WrittenBy table:

Functional Dependencies:
(Book_id, Author_id) -> No additional functional dependencies beyond the primary key.

11)PUBLISHER table:

Functional Dependencies:
Pub_id -> Pub_name, Pub_country

12)SALES table:

Functional Dependencies:
(Book_id, Vendor_id) -> No additional functional dependencies beyond the primary key.

05

SQL code

(with Triggers concept & also with 7 created Queries)

Library table

```
CREATE TABLE LIBRARY (
  Lib_phno VARCHAR(10),
  Lib_name VARCHAR(50),
  Lib_id INT PRIMARY KEY,
  Lib_address_x FLOAT,
  Lib_address_y FLOAT
);
```

	LIB_PHNO	LIB_NAME	LIB_ID	LIB_ADDRESS_X	LIB_ADDRESS_Y
1	1234567890	Library A	1	12.345	67.89
2	9876543210	Library B	2	98.765	43.21
3	5678901234	Library C	3	56.789	90.123
4	4321098765	Library D	4	43.21	98.765
5	1111111111	Library E	5	11.111	11.111
6	2222222222	Library F	6	22.222	22.222
7	3333333333	Library G	7	33.333	33.333
8	4444444444	Library H	8	44.444	44.444
9	5555555555	Library I	9	55.555	55.555
10	6666666666	Library J	10	66.666	66.666

books table

```
CREATE TABLE BOOKS (
  Count INT,
  Book_id INT PRIMARY KEY,
  Book_status VARCHAR(50),
  Book_price DECIMAL(10, 2),
  Book_title VARCHAR(100),
  Lib_id INT,
  Pub_id INT,
  FOREIGN KEY (Lib_id) REFERENCES
LIBRARY (Lib_id),
  FOREIGN KEY (Pub_id) REFERENCES
PUBLISHER (Pub_id)
);
```

	COUNT	BOOK_ID	BOOK_STATUS	BOOK_PRICE	BOOK_TITLE	LIB_ID	PUB_ID
1	5	1	Available	10.99	Book A	(null)	(null)
2	3	2	Unavailable	15.99	Book B	(null)	(null)
3	7	3	Available	12.99	Book C	(null)	(null)
4	2	4	Available	9.99	Book D	(null)	(null)
5	9	5	Unavailable	14.99	Book E	(null)	(null)
6	4	6	Available	11.99	Book F	(null)	(null)
7	6	7	Unavailable	13.99	Book G	(null)	(null)
8	8	8	Available	16.99	Book H	(null)	(null)
9	1	9	Available	8.99	Book I	(null)	(null)
10	7	10	Available	17.99	Book J	(null)	(null)

Admin table

```
CREATE TABLE ADMIN (
  Admin_id INT PRIMARY KEY,
  Admin_name VARCHAR(50),
  Admin_role VARCHAR(50)
);
```

	ADMIN_ID	ADMIN_NAME	ADMIN_ROLE
1	1	John Doe	Administrator
2	2	Jane Smith	Supervisor
3	3	Mike Johnson	Manager
4	4	Emily Davis	Coordinator
5	5	Robert Wilson	Assistant
6	6	Roshan chowdary	Assistant
7	7	Kenzo Tenma	Assistant
8	8	Johan Denma	Assistant
9	9	williams Smith	Assistant
10	10	Warner Doe	Assistant

Employee table

```
CREATE TABLE EMPLOYEE (
  Emp_id INT PRIMARY KEY,
  Emp_name VARCHAR(50),
  Emp_position VARCHAR(50),
  Emp_dept VARCHAR(50)
);
```

	EMP_ID	EMP_NAME	EMP_POSITION	EMP_DEPT
1	1	John Smith	Manager	Finance
2	2	Jane Johnson	Engineer	Engineering
3	3	Michael Brown	Analyst	Marketing
4	4	Emily Davis	Designer	Creative
5	5	Robert Wilson	Developer	IT
6	6	Jennifer Thompson	Sales Representative	Sales
7	7	David Martinez	HR Manager	Human Resources
8	8	Jessica Lee	Project Manager	Operations
9	10	Sarah Walker	Customer Support	Customer Service
10	9	Daniel Anderson	Researcher	R1

VENDOR table

```
CREATE TABLE VENDOR (
  Vendor_id INT PRIMARY KEY,
  Vendor_name VARCHAR(50)
);
```

	VENDOR_ID	VENDOR_NAME
1	1	Ramesh
2	2	Suresh
3	3	Ganesh
4	4	Jayesh
5	5	Bhuvanesh

RECORDS table

```
CREATE TABLE RECORDS (
  Received_date DATE,
  issued_date DATE,
  Req_id INT PRIMARY KEY,
  fine DECIMAL(10, 2),
  Book_id INT,
  Mem_id INT,
  Emp_id INT,
  FOREIGN KEY (Book_id) REFERENCES
BOOKS (Book_id),
  FOREIGN KEY (Mem_id) REFERENCES
MEMBER (Mem_id),
  FOREIGN KEY (Emp_id) REFERENCES
EMPLOYEE (Emp_id)
);
```

	RECEIVED_DATE	ISSUED_DATE	REQ_ID	FINE	BOOK_ID	MEM_ID	EMP_ID
1	02-01-23	06-01-23	1	0	1	1	1
2	03-02-23	07-02-23	2	0	2	2	2
3	04-03-23	08-03-23	3	0	3	3	3
4	05-04-23	09-04-23	4	0	4	4	4
5	06-05-23	10-05-23	5	0	5	5	5
6	07-06-23	11-06-23	6	0	6	6	6
7	08-07-23	12-07-23	7	0	7	7	7
8	09-08-23	13-08-23	8	0	8	8	8
9	10-09-23	14-09-23	9	0	9	9	9
10	11-10-23	15-10-23	10	0	10	10	10
11	(null)	15-10-23	11	0	10	10	10

MEMBER table

```
CREATE TABLE MEMBER (
  Mem_id INT PRIMARY KEY,
  Mem_phno VARCHAR(10),
  DOB DATE,
  Mem_type VARCHAR(50),
  Mem_first_name VARCHAR(50),
  Mem_last_name VARCHAR(50),
  Mem_address_x NUMBER,
  Mem_address_y NUMBER
);
```

	MEM_ID	MEM_PHNO	DOB	MEM_TYPE	MEM_FIRST_NAME	MEM_LAST_NAME	MEM_ADDRESS_X	MEM_ADDRESS_Y
1	1	1234567890	01-01-90	B_tech	John	Smith	12.345	67.89
2	2	9876543210	15-03-92	M_tech	Jane	Johnson	98.765	43.21
3	3	5678901234	10-07-95	MCA	Michael	Brown	56.789	90.123
4	4	4321098765	05-05-98	PHD	Emily	Davis	43.21	98.765
5	5	1111111111	11-11-93	B_com	Robert	Wilson	11.111	11.111
6	6	2222222222	16-06-96	MSC	Jennifer	Thompson	22.222	22.222
7	7	3333333333	20-09-94	B_tech	David	Martinez	33.333	33.333
8	8	4444444444	25-12-97	MCA	Jessica	Lee	44.444	44.444
9	9	5555555555	30-04-91	M_tech	Daniel	Anderson	55.555	55.555
10	10	6666666666	08-08-99	B_com	Sarah	Walker	66.666	66.666

Book table

```
CREATE TABLE book (
  Book_id INT,
  Req_id INT,
  PRIMARY KEY (Book_id, Req_id),
  FOREIGN KEY (Book_id) REFERENCES BOOKS
(Book_id),
  FOREIGN KEY (Req_id) REFERENCES RECORDS
(Req_id)
);
```

	BOOK_ID	REQ_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10

AUTHOR table

```
CREATE TABLE AUTHOR (
  Author_id INT PRIMARY KEY,
  qualification VARCHAR(50),
  Author_name VARCHAR(100),
  Author_subject VARCHAR(100)
);
```

	AUTHOR_ID	QUALIFICATION	AUTHOR_NAME	AUTHOR_SUBJECT
1	1	PhD	John Smith	Computer Science
2	2	MSc	Jane Johnson	Mathematics
3	3	MA	Michael Brown	History
4	4	PhD	Emily Davis	Psychology
5	5	MSc	Robert Wilson	Physics
6	6	MA	Jennifer Thompson	English Literature
7	7	PhD	David Martinez	Sociology
8	8	MSc	Jessica Lee	Chemistry
9	9	MA	Daniel Anderson	Economics
10	10	PhD	Sarah Walker	Biology

WrittenBy table

```
CREATE TABLE WrittenBy (
  Book_id INT,
  Author_id INT,
  PRIMARY KEY (Book_id, Author_id),
  FOREIGN KEY (Book_id) REFERENCES
BOOKS (Book_id),
  FOREIGN KEY (Author_id) REFERENCES
AUTHOR (Author_id)
);
```

	BOOK_ID	AUTHOR_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10

PUBLISHER table

```
CREATE TABLE PUBLISHER (
  Pub_id INT PRIMARY KEY,
  Pub_name VARCHAR(100),
  Pub_country VARCHAR(100)
);
```

	PUB_ID	PUB_NAME	PUB_COUNTRY
1	1	Publisher A	India
2	2	Publisher B	Australia
3	3	Publisher C	France
4	4	Publisher D	China
5	5	Publisher E	Serbia
6	6	Publisher F	France
7	7	Publisher G	India
8	8	Publisher H	Australia
9	9	Publisher I	Serbia
10	10	Publisher J	India

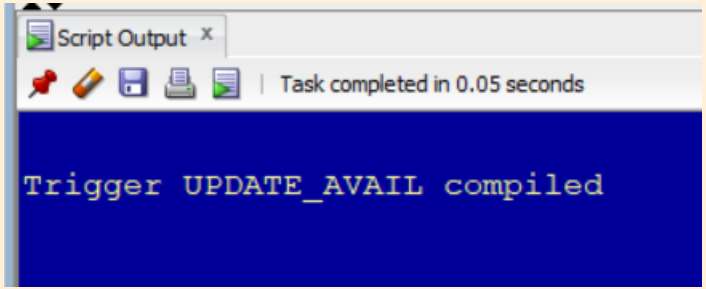
SALES table

```
CREATE TABLE SALES (
  Book_id INT,
  Vendor_id INT,
  PRIMARY KEY (Book_id, Vendor_id),
  FOREIGN KEY (Book_id) REFERENCES BOOKS
(Book_id),
  FOREIGN KEY (Vendor_id) REFERENCES AUTHOR
(Vendor_id)
);
```

	BOOK_ID	VENDOR_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

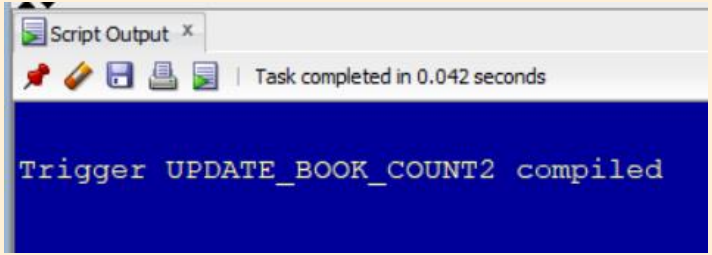
Trigger to update the available books

```
CREATE OR REPLACE TRIGGER UPDATE_AVAIL
BEFORE INSERT OR UPDATE ON RECORDS
FOR EACH ROW
DECLARE
   UNAVAIL_COUNT NUMBER;
   STORED_COUNT NUMBER;
BEGIN
   SELECT COUNT(*) INTO UNAVAIL_COUNT FROM RECORDS
   WHERE BOOK_ID = :NEW.BOOK_ID AND RECEIVED_DATE IS NULL;
   SELECT COUNT(*) INTO STORED_COUNT FROM BOOKS
   WHERE BOOK_ID = :NEW.BOOK_ID;
   IF UNAVAIL_COUNT = STORED_COUNT THEN
      UPDATE BOOKS
      SET BOOK_STATUS = 'Unavailable'
      WHERE BOOK_ID = :NEW.BOOK_ID;
      RAISE_APPLICATION_ERROR(-20000, 'BOOK NOT AVAILABLE');
   END IF;
   IF UNAVAIL_COUNT + 1 <= STORED_COUNT THEN
      UPDATE BOOKS
      SET BOOK_STATUS = 'Available'
      WHERE BOOK_ID = :NEW.BOOK_ID;
   END IF;
END;
/
```



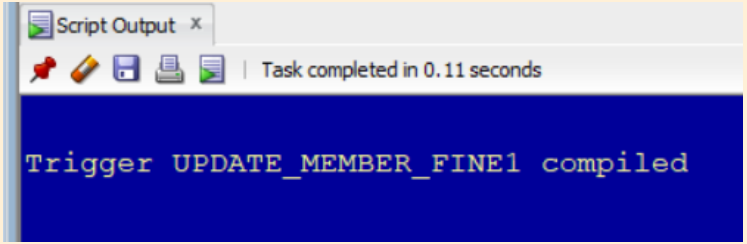
Trigger to update the books count

```
CREATE OR REPLACE TRIGGER update_book_count2
AFTER INSERT ON RECORDS
FOR EACH ROW
BEGIN
  UPDATE BOOKS
  SET Count = Count - 1
  WHERE Book_id = :NEW.Book_id;
END;
/
```



Trigger to update the member fine

```
CREATE OR REPLACE TRIGGER update_member_fine1
AFTER INSERT ON RECORDS
FOR EACH ROW
  DECLARE total_fine DECIMAL(10, 2);
    BEGIN
  -- Calculate the total fine for the member
  SELECT SUM(fine) INTO total_fine
  FROM RECORDS
  WHERE Mem_id = :NEW.Mem_id;
  -- Update the member's fine in the MEMBER table
  UPDATE RECORDS
  SET fine = total_fine
  WHERE Mem_id = :NEW.Mem_id;
END;
/
```



1)What are the books written by Author John Smith

```
SELECT b.Book_title, a.Author_name
FROM BOOKS b
JOIN WrittenBy w ON b.Book_id = w.Book_id
JOIN AUTHOR a ON w.Author_id = a.Author_id
WHERE a.Author_name = 'John Smith’;
```

Query Result

SQL | All Rows Fetched: 1 in 0.039 seconds

	BOOK_TITLE	AUTHOR_NAME
1	Book A	John Smith

2)What are the books written by Authors who have a qualification of PhD

```
SELECT b.Book_title, a.Author_name
FROM BOOKS b
JOIN WrittenBy wb ON b.Book_id = wb.Book_id
JOIN AUTHOR a ON wb.Author_id = a.Author_id
WHERE a.qualification = 'PhD';
```

Query Result

All Rows Fetched: 4 in 0.008 seconds

	BOOK_TITLE	AUTHOR_NAME
1	Book A	John Smith
2	Book D	Emily Davis
3	Book G	David Martinez
4	Book J	Sarah Walker

3) What is the total sales amount for each vendor

```
SELECT v.Vendor_id, v.Vendor_name, SUM(b.Book_price) AS Total
FROM VENDOR v
JOIN SALES s ON v.Vendor_id = s.Vendor_id
JOIN BOOKS b ON s.Book_id = b.Book_id
GROUP BY v.Vendor_id, v.Vendor_name;
```

Query Result

All Rows Fetched: 5 in 0.005 seconds

	VENDOR_ID	VENDOR_NAME	TOTAL
1	2	Suresh	15.99
2	4	Jayesh	9.99
3	1	Ramesh	10.99
4	3	Ganesh	12.99
5	5	Bhuvanesh	14.99

4)What are the books sold by vendor Bhuvanesh

```
SELECT b.Book_title, v.Vendor_name
FROM BOOKS b
JOIN SALES s ON b.Book_id = s.Book_id
JOIN VENDOR v ON s.Vendor_id = v.Vendor_id
WHERE v.Vendor_name = 'Bhuvanesh';
```

Query Result

SQL | All Rows Fetched: 1 in 0.

	BOOK_TITLE	VENDOR_NAME
1	Book E	Bhuvanesh

5) Obtain all the records with respective book, member, employee information

```
SELECT r.Req_id, r.Received_date, r.issued_date, r.fine,
       b.Book_title, m.Mem_first_name, m.Mem_last_name, e.Emp_name
FROM RECORDS r
JOIN BOOKS b ON r.Book_id = b.Book_id
JOIN MEMBER m ON r.Mem_id = m.Mem_id
JOIN EMPLOYEE e ON r.Emp_id = e.Emp_id;
```

Query Result

All Rows Fetched: 11 in 0.023 seconds

	REQ_ID	RECEIVED_DATE	ISSUED_DATE	FINE	BOOK_TITLE	MEM_FIRST_NAME	MEM_LAST_NAME	EMP_NAME
1	1	02-01-23	06-01-23	0	Book A	John	Smith	John Smith
2	2	03-02-23	07-02-23	0	Book B	Jane	Johnson	Jane Johnson
3	3	04-03-23	08-03-23	0	Book C	Michael	Brown	Michael Brown
4	4	05-04-23	09-04-23	0	Book D	Emily	Davis	Emily Davis
5	5	06-05-23	10-05-23	0	Book E	Robert	Wilson	Robert Wilson
6	6	07-06-23	11-06-23	0	Book F	Jennifer	Thompson	Jennifer Thompson
7	7	08-07-23	12-07-23	0	Book G	David	Martinez	David Martinez
8	8	09-08-23	13-08-23	0	Book H	Jessica	Lee	Jessica Lee
9	10	11-10-23	15-10-23	0	Book J	Sarah	Walker	Sarah Walker
10	11	(null)	15-10-23	0	Book J	Sarah	Walker	Sarah Walker
11	9	10-09-23	14-09-23	0	Book I	Daniel	Anderson	Daniel Anderson

6) What are the books published by Publisher A

```
SELECT b.Book_title, p.Pub_name
FROM BOOKS b
JOIN PUBLISHER p ON b.Book_id = p.Pub_id
WHERE p.Pub_name = 'Publisher A';
```

Query Result

SQL | All Rows Fetched: 1 in 0.005 seconds

	BOOK_TITLE	PUB_NAME
1	Book A	Publisher A

7) What is the total fine collected by each member

```
SELECT m.Mem_id, m.Mem_first_name, m.Mem_last_name, SUM(r.fine) AS Total
FROM MEMBER m
JOIN RECORDS r ON m.Mem_id = r.Mem_id
GROUP BY m.Mem_id, m.Mem_first_name, m.Mem_last_name;
```

Query Result

All Rows Fetched: 10 in 0.003 seconds

	MEM_ID	MEM_FIRST_NAME	MEM_LAST_NAME	TOTAL
1	3	Michael	Brown	0
2	4	Emily	Davis	0
3	5	Robert	Wilson	0
4	10	Sarah	Walker	0
5	9	Daniel	Anderson	0
6	6	Jennifer	Thompson	0
7	7	David	Martinez	0
8	1	John	Smith	0
9	2	Jane	Johnson	0
10	8	Jessica	Lee	0

Our team

Neerati Bhuvanesh

Section-A

Roll.no- 21CSB0A39

Peram Abhishek

Section-A

Roll.no- 21CSB0A42

Thank You

Does anyone have
any questions?

nb21csb0a39@student.nitw.ac.in
pa21csb0a42@student.nitw.ac.in

+91 7032726969
+91 6281616929