Cranes Varsity

'Where Technology Meets Excellence'

Name:	**Maharshi Goswami**
College Reg.No	**U03KW21S0025**
Batch NO:	**PGDDM46**
Cranes Reg. No	**R202410045261505**
Project Title:	**RDBMS usingSQL**
S/w Used:	**SQL Workbench**
Trainer Name:	**Sir Basavaraj**

Abstract:

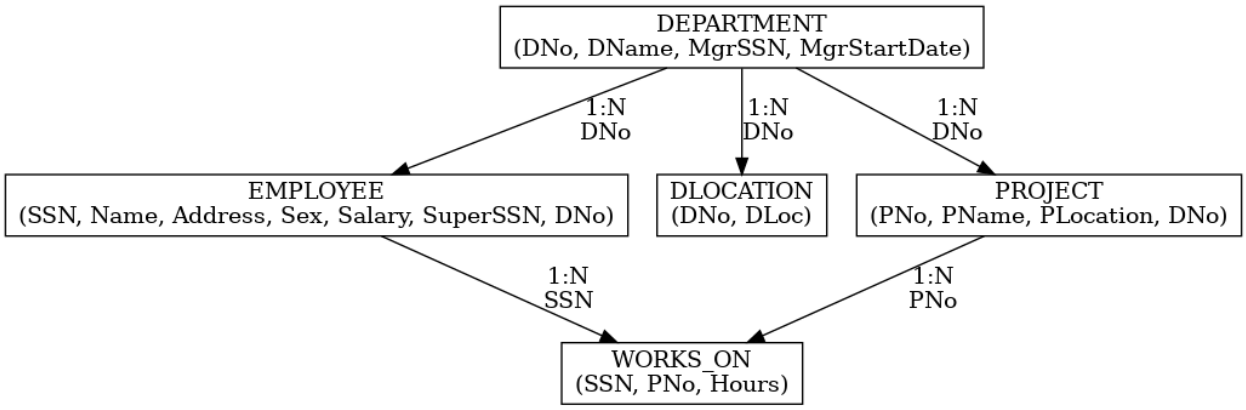
This project involves the design, implementation, and optimization of SQL queries for multiple database systems, including Company, Library, Movie, Order, and College databases. The objective is to demonstrate proficiency in database management and SQL operations by solving real-world problems related to employee management, academic performance tracking, library book lending, movie production, and sales order processing. The project encompasses the creation of relational database structures, the manipulation and retrieval of data using advanced SQL queries, and the application of techniques such as aggregation, subqueries, joins, and views. Through tasks such as updating employee salaries, managing library inventories, calculating student performance, and handling sales transactions, the project showcases the use of SQL in effectively managing and analyzing large datasets. The knowledge and skills gained from this project emphasize the practical applications of SQL in diverse business and academic environments, laying a strong foundation for database management roles in the industry.

Introduction to the Project:

The aim of this project is to develop and implement SQL-based solutions for various real-world scenarios by designing and managing databases in multiple domains, including company management, library systems, movie production, order processing, and academic performance tracking. This project focuses on creating relational databases and applying advanced SQL operations to retrieve, manipulate, and analyze data efficiently. Through the use of SQL queries, the project addresses key business needs such as employee management, inventory control, academic performance evaluation, and sales order processing. The scope of the project covers defining database schemas, creating tables, inserting data, and performing complex queries, which include aggregations, subqueries, joins, and other advanced SQL techniques. By building databases for these diverse systems, this project not only enhances the practical understanding of database management but also prepares the groundwork for real-world applications, making it suitable for roles in database administration, business intelligence, and data analytics.

Project 1: Company Database Management System

ER Diagram:



The ER diagram for the **Company Database** provides a clear and organized representation of the database structure, highlighting the entities, their attributes, and the relationships between them. The diagram includes five primary entities: **EMPLOYEE**, **DEPARTMENT**, **DLOCATION**, **PROJECT**, and **WORKS_ON**. Each table is uniquely identified by a primary key, ensuring data integrity, and foreign keys are used to establish relationships between the tables. For example, the **EMPLOYEE** table references the **DEPARTMENT** table through the DNo attribute, linking employees to their respective departments. Similarly, the **PROJECT** table is associated with the **DEPARTMENT** table through DNo, indicating which department controls a specific project. The **WORKS_ON** table serves as a junction table, connecting employees to the projects they work on, and includes a composite primary key (SSN, PNo) to ensure uniqueness.

Table Descriptions and Dataset

1. EMPLOYEE

Columns:

- SSN (Primary Key)
- Ename
- Address
- Sex
- Salary
- SuperSSN (Foreign Key)
- DNo (Foreign Key)

Employee's Sample Data:

SSN	Ename	Address	Sex	Salary	SuperSSN	DNo
111111111	Mike Scott	123A	M	700000.00		1
222222222	Jane Doe	BT	F	650000.00	111111111	2

2. DEPARTMENT

Columns:

- DNo (Primary Key)
- DName
- MgrSSN (Foreign Key)
- MgrStartDate

DEPARTMENT'S Sample Data:

DNo	DName	MgrSSN	MgrStartDate
1	**Account**	**111111111**	**2024-09-25**
2	**HR**	**222222222**	**2019-03-01**

3.DLOCATION

Columns:

- DNo
- DLOC

DLOCATION'S Sample Data:

DNo	DLoc
1	NewYork
2	ScotLand

4.PROJECT

Columns:

- PNo(primary key)
- PName
- PLocation
- DNo(Foreign key)

Project's Sample Data:

PNo	PName	PLocation	DNo
101	IOT	London	5
102	Cloud Migration	San Francisco	3

5.WORKS ON

Columns:

- SSN(ForeignKey)
- PNo(Foreign Key)
- Hours

Works On Sample Data:

SSN	PNo	Hours
111111111	101	20:00
222222222	102	15:00

Query1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project.

```
SELECT DISTINCT P.PNo
FROM PROJECT P
JOIN DEPARTMENT D ON P.DNo = D.DNo
LEFT JOIN EMPLOYEE E ON E.DNo = P.DNo OR E.SSN = P.PNo
WHERE (E.Ename LIKE '%Scott%' OR D.MgrSSN = E.SSN);
```

Result Grid

PNo
103
102

Query2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.

```
SELECT E.Ename, E.Salary, E.Salary * 1.10 AS NewSalary
FROM EMPLOYEE E
JOIN WORKS_ON W ON E.SSN = W.SSN
JOIN PROJECT P ON W.PNo = P.PNo
WHERE P.PName = 'IoT';
```

	Ename	Salary	NewSalary
▶	Mike Scott	700000.00	770000.0000
	Trixie Scott	500000.00	550000.0000
	Tim Lee	550000.00	605000.0000

Query3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department.

```
SELECT
SUM(E.Salary) AS TotalSalaries,
MAX(E.Salary) AS MaxSalary,
MIN(E.Salary) AS MinSalary,
AVG(E.Salary) AS AvgSalary
FROM EMPLOYEE E
JOIN DEPARTMENT D ON E.DNo = D.DNo
WHERE D.DName = 'Accounts';
```

	TotalSalaries	MaxSalary	MinSalary	AvgSalary
▶	1200000.00	700000.00	500000.00	600000.000000

Query4. Retrieve the name of each employee who works on all the projects controlled by department number 4 (use NOT EXISTS operator). For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.

```
SELECT E.Ename
FROM EMPLOYEE E
WHERE NOT EXISTS (
SELECT P.PNo
FROM PROJECT P
WHERE P.DNo = 4
AND NOT EXISTS (
SELECT W.SSN
FROM WORKS_ON W
WHERE W.PNo = P.PNo
AND W.SSN = E.SSN
)
);
```

Ename
Mike Scott
Jane Doe
Robert Smith
Trixie Scott
Michael Jackson
Tim Lee

Project 2: Library Database Management System

ER Diagram:

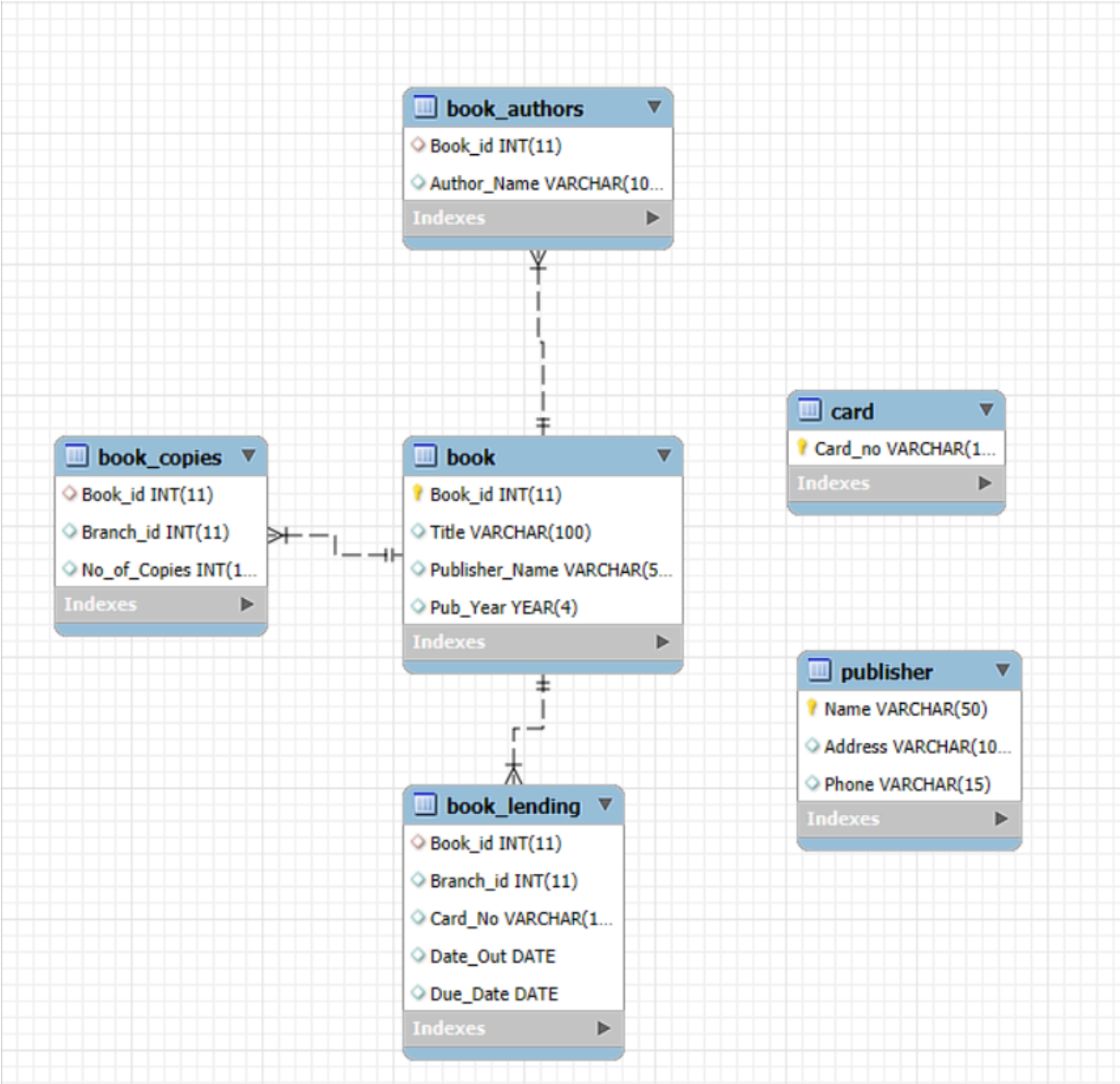


Table Descriptions and Dataset

BOOK

Columns:

- Book_id (Primary Key):
- Title:
- Publisher_Name:
- Pub_Year:

Book's sample data:

BOOK_ID	Title	Publisher_name	Pub_Year
1	Indian Culture	Publisher A	2016
2	Modern India	Publisher B	2017

BOOK_AUTHORS Table

Columns:

- Book_id (Foreign Key):
- Author_Name:

Book_Author's sample data:

Book_ID	Author_Name
1	Ravi Sharma
2	Anil Gupta

Publisher Table

Columns:

- Name (Primary Key)
- Address
- Phone

Publisher's sample data:

Name	Address	Phone
Publisher A	Mumbai, India	9876543210
Publisher B	Delhi, India	9876543211

BOOK_COPIES

Columns:

- Book_id (Foreign Key)
- Branch_id
- No_of_Copies
- Composite Primary Key: Book_id and Branch_id.

Book_Copies sample data:

Book_id	Branch_id	Card_No
1	1	10
1	2	15

BOOK LENDING TABLE

Columns:

- Book_id (Foreign Key)
- Branch_id (Foreign Key):
- Card_No
- Date_Out
- Due_Date
- Composite Primary Key: Book_id and Branch_id.

Book Lending sample data :

Book_id	Branch_id	Card_No	Date_Out	Due_Date
1	1	C007	2017-01-10	2017-01-20
1	2	C006	2017-05-05	2017-05-15

LIBRARY BRANCH TABLE

Columns:

- Branch_id (Primary Key)
- Branch_Name
- Address

Branch_id	**Branch_Name**	**Address**
1	Mumbai Branch	Mumbai,India
2	Delhi Branch	Delhi,India

CARD TABLE

Columns:

- Card_No(Primary Key)

Card_No Sample Data:

Card_No
C001
C002
C003

Query 1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.

```
SELECT B.Book_id, B.Title, B.Publisher_Name, B.Pub_Year, A.Author_Name,
    BC.Branch_id, BC.No_of_Copies
FROM BOOK B
JOIN BOOK_AUTHORS A ON B.Book_id = A.Book_id
JOIN BOOK_COPIES BC ON B.Book_id = BC.Book_id;
```

Book_id	Title	Publisher_Name	Pub_Year	Author_Name	Branch_id	No_of_Copies
1	Indian Culture	Publisher A	2016	Ravi Sharma	1	10
1	Indian Culture	Publisher A	2016	Ravi Sharma	2	15
2	Modern India	Publisher B	2017	Anil Gupta	1	5
3	History of India	Publisher A	2015	Kiran Mehta	2	7
4	Indian Architecture	Publisher C	2018	Sunita Desai	1	12
5	Indian Festivals	Publisher A	2019	Rajesh Kumar	2	8

Query 2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017

```
SELECT BL.Card_No, COUNT(*) AS Total_Books_Borrowed
FROM BOOK_LENDING BL
WHERE BL.Date_Out BETWEEN '2017-01-01' AND '2017-06-30'
GROUP BY BL.Card_No
HAVING COUNT(*) > 3;
```

	Card_No	Total_Books_Borrowed
▶	C001	5

Query 3. Delete a book in the BOOK table. Update the contents of other tables to reflect this data manipulation operation.

```
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;
```

Query 4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

```
CREATE VIEW available_books AS

SELECT B.Book_id, B.Title, B.Publisher_Name, BC.Branch_id, BC.No_of_Copies

FROM BOOK B

JOIN BOOK_COPIES BC ON B.Book_id = BC.Book_id

WHERE BC.No_of_Copies > 0;

select*from available_books;
```

Book_id	Title	Publisher_Name	Branch_id	No_of_Copies
1	Indian Culture	Publisher A	1	10
1	Indian Culture	Publisher A	2	15
2	Modern India	Publisher B	1	5
3	History of India	Publisher A	2	7
4	Indian Architecture	Publisher C	1	12
5	Indian Festivals	Publisher A	2	8

Project 3: Movie Database Management System

ER Diagram:

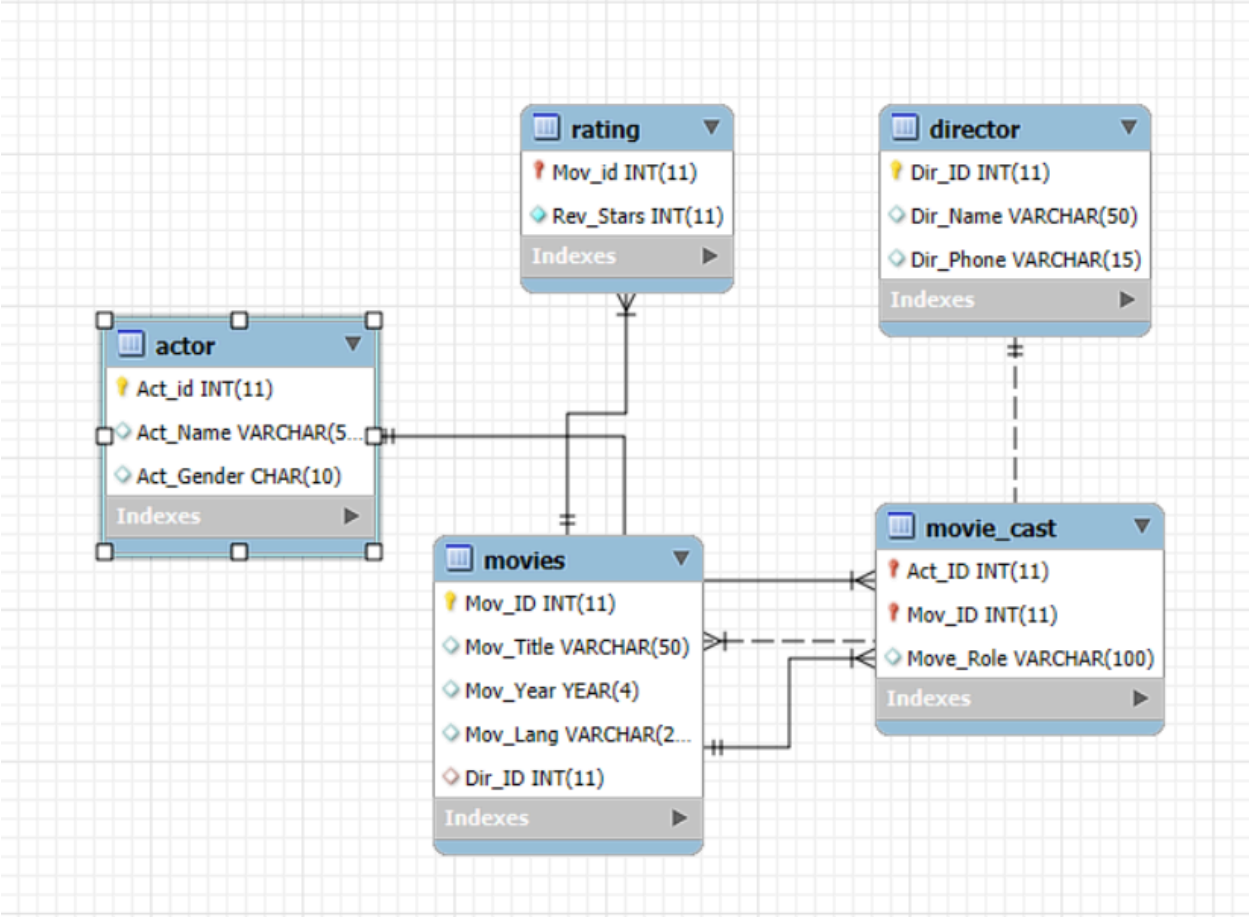


Table Descriptions and Dataset

ACTOR

- Act_id (INT, Primary Key)
- Act_Name (VARCHAR(50))
- Act_Gender (CHAR(10)).

Actor Sample Data:

Act_id	Act_Name	Act_Gender
1	Tom Hanks	Male
2	Leonardo DiCaprio	Male

DIRECTOR

- Dir_id (INT, Primary Key)
- Dir_Name (VARCHAR(50))
- Dir_Phone (VARCHAR(15))
-

Director's Sample Data:

Dir_ID	Dir_Name	Dir_Phone
1	Hitchcock	1234567890
2	Steven Spielberg	0987654321

MOVIES

- Mov_id (INT, Primary Key):
- Mov_Title (VARCHAR(50)):
- Mov_Year (YEAR):
- Mov_Lang (VARCHAR(20)):
- Dir_id (INT, Foreign Key):

Movies Sample Data:

Mov_ID	Mov_Title	Mov_year	Mov_Lang	Dir_ID
101	Psycho	1960	English	1
102	Vertigo	1958	English	1

MOVIE_CAST

- Act_id (INT, Foreign Key):
- Mov_id (INT, Foreign Key):
- Role (VARCHAR(100)):
- Composite Primary Key: (Act_id, Mov_id).

Movie_Cast Sample Data:

Act_ID	Mov_ID	Movie_Role
1	101	Protagonist
1	106	Lead

RATING

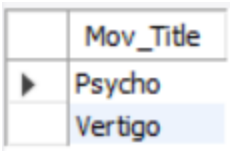
- `Mov_id` (INT, Primary Key, Foreign Key)
- `Rev_Stars` (INT, NOT NULL)

Rating Sample Data:

Mov_ID	Rev_Stars
101	5
102	4

Query 1. List the titles of all movies directed by 'Hitchcock'.

```
SELECT M.Mov_Title
FROM MOVIES M
JOIN DIRECTOR D ON M.Dir_ID = D.Dir_ID
WHERE D.Dir_Name = 'Hitchcock';
```



	Mov_Title
▶	Psycho
	Vertigo

Query 2. Find the movie names where one or more actors acted in two or more movies.

```
SELECT M.Mov_Title
FROM MOVIES M
JOIN MOVIE_CAST MC ON M.Mov_ID = MC.Mov_ID
GROUP BY M.Mov_Title
HAVING COUNT(MC.Act_ID) >= 2;
```

Mov_Title
Tenet

Query 3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

```
#Actors in movies before 2000:
SELECT A.Act_Name, M.Mov_Title, M.Mov_Year
FROM ACTOR A
JOIN MOVIE_CAST MC ON A.Act_ID = MC.Act_ID
JOIN MOVIES M ON MC.Mov_ID = M.Mov_ID
WHERE M.Mov_Year < 2000;

#Actors in movies after 2015:
SELECT A.Act_Name, M.Mov_Title, M.Mov_Year
FROM ACTOR A
JOIN MOVIE_CAST MC ON A.Act_ID = MC.Act_ID
```

```
JOIN MOVIES M ON MC.Mov_ID = M.Mov_ID
WHERE M.Mov_Year > 2015;
```

Act_Name	Mov_Title	Mov_Year
Tom Hanks	Psycho	1960
Tom Hanks	Tenet	2020
Scarlett Johansson	Tenet	2020

Query 4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

```
SELECT M.Mov_Title, R.Rev_Stars
FROM MOVIES M
JOIN RATING R ON M.Mov_ID = R.Mov_ID
WHERE R.Rev_Stars IS NOT NULL
GROUP BY M.Mov_Title
ORDER BY M.Mov_Title;
```

Mov_Title	Rev_Stars
Inception	4
Jaws	5
Psycho	5
Schindler's List	5
Tenet	5
Vertigo	4

Query 5. Update rating of all movies directed by 'Steven Spielberg' to 5.

```
UPDATE RATING
SET Rev_Stars = 5
WHERE Mov_id IN (
    SELECT Mov_id
    FROM MOVIES
    JOIN DIRECTOR ON MOVIES.Dir_id = DIRECTOR.Dir_id
    WHERE Dir_Name = 'Steven Spielberg'
);
 select*from RATING;
```

Mov_id	Rev_Stars
101	5
102	4
103	5
104	5
105	4
106	5

Project 4: Order Database Management System

ER Diagram:

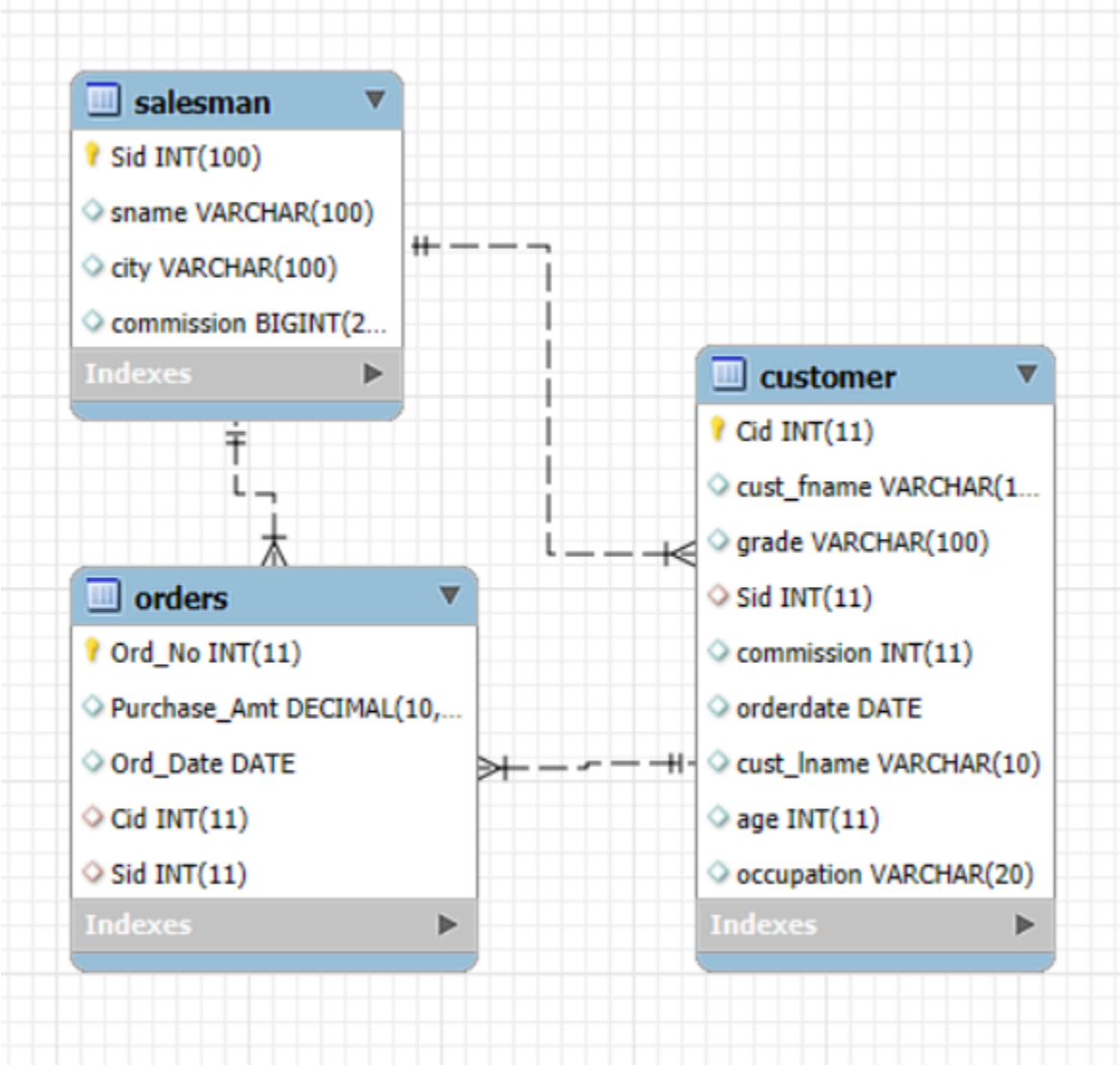


Table Descriptions and Dataset

SALESMAN

- Salesman_id (INT, Primary Key)
- Name (VARCHAR(50))
- City (VARCHAR(50))
- Commission (DECIMAL(5, 2))

Salesman Sample Data:

Sid	sname	city	commission
1	virat	Bihar	26
2	Suraj	Ranchi	

CUSTOMER

- Customer_id (INT, Primary Key)
- Cust_Name (VARCHAR(50))
- City (VARCHAR(50))
- Grade (INT)
- Salesman_id (INT, Foreign Key)

Customer Sample Data:

Cid	cust_fname	grade	Sid	commission	orderdate	cust_lname	age	occupation
1	ffjr	B	2	7	2024-07-13	sdhaks	20	actor
2	edheu	A	3	6	2024-04-12	dddjs	30	actor

ORDERS

- Ord_No (INT, Primary Key)
- Purchase_Amt (DECIMAL(10, 2))
- Ord_Date (DATE)
- Customer_id (INT, Foreign Key)
- Salesman_id (INT, Foreign Key)

Order's Sample Data:

Ord_No	Purchase_Amt	Ord_Date	Cid	Sid
1	150.00	2024-07-13	1	2
2	200.00	2024-04-12	2	3

Query 1. Count the customers with grades above 's average.

```
SELECT COUNT(*) AS Customers_Above_Average_Grade

FROM CUSTOMER

WHERE

   Grade > (SELECT AVG(

            CASE

               WHEN Grade = 'A' THEN 3

               WHEN Grade = 'B' THEN 2

               WHEN Grade = 'C' THEN 1

               ELSE 0

            END) AS avg_grade
```

```
    FROM CUSTOMER);
```

Query 2. Find the name and numbers of all salesmen who had more than one customer.

```
SELECT S.sname, S.Sid
FROM Salesman S
INNER JOIN CUSTOMER C ON S.Sid = C.Sid
GROUP BY S.sname, S.Sid
HAVING COUNT(C.Cid) > 1;
```

	sname	Sid
▶	varun	3

Query 3. List all salesmen and indicate those who have and don't have customers in their cities

```
SELECT S.sname, S.Sid, 'Has Customers' AS Status
FROM Salesman S
INNER JOIN CUSTOMER C ON S.Sid = C.Sid
WHERE S.city = C.city

UNION

SELECT S.sname, S.Sid, 'No Customers' AS Status
```

```
FROM Salesman S
LEFT JOIN CUSTOMER C ON S.Sid = C.Sid
WHERE C.Sid IS NULL OR S.city != C.city;
select*from salesman;
select*from customer;
select*from orders;
```

Ord_No	Purchase_Amt	Ord_Date	Cid	Sid
1	150.00	2024-07-13	1	2
2	200.00	2024-04-12	2	3
3	350.00	2023-04-12	3	3
4	450.00	2024-12-24	4	4

Query 4. Create a view that finds the salesman who has the customer with the highest order of a day.

```
CREATE OR REPLACE VIEW Top_Salesman_View AS
SELECT S.sname, S.Sid, C.cust_fname, C.cust_lname, O.Purchase_Amt, O.Ord_Date
FROM Salesman S
INNER JOIN ORDERS O ON S.Sid = O.Sid
INNER JOIN CUSTOMER C ON O.Cid = C.Cid
WHERE O.Purchase_Amt = (SELECT MAX(Purchase_Amt) FROM ORDERS WHERE
Ord_Date = O.Ord_Date);
```

select*from Top_Salesman_View;

sname	Sid	cust_fname	cust_lname	Purchase_Amt	Ord_Date
suraj	2	ffjr	sdhaks	150.00	2024-07-13
varun	3	edheu	dddjs	200.00	2024-04-12
varun	3	ekfhei	dddasj	350.00	2023-04-12
kunal	4	jjfjfe	kaoops	450.00	2024-12-24

Query 5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

DELETE FROM ORDERS

WHERE Sid = 101;

DELETE FROM Salesman

WHERE Sid = 101;

Project 5: College Database Management System

ER Diagram:

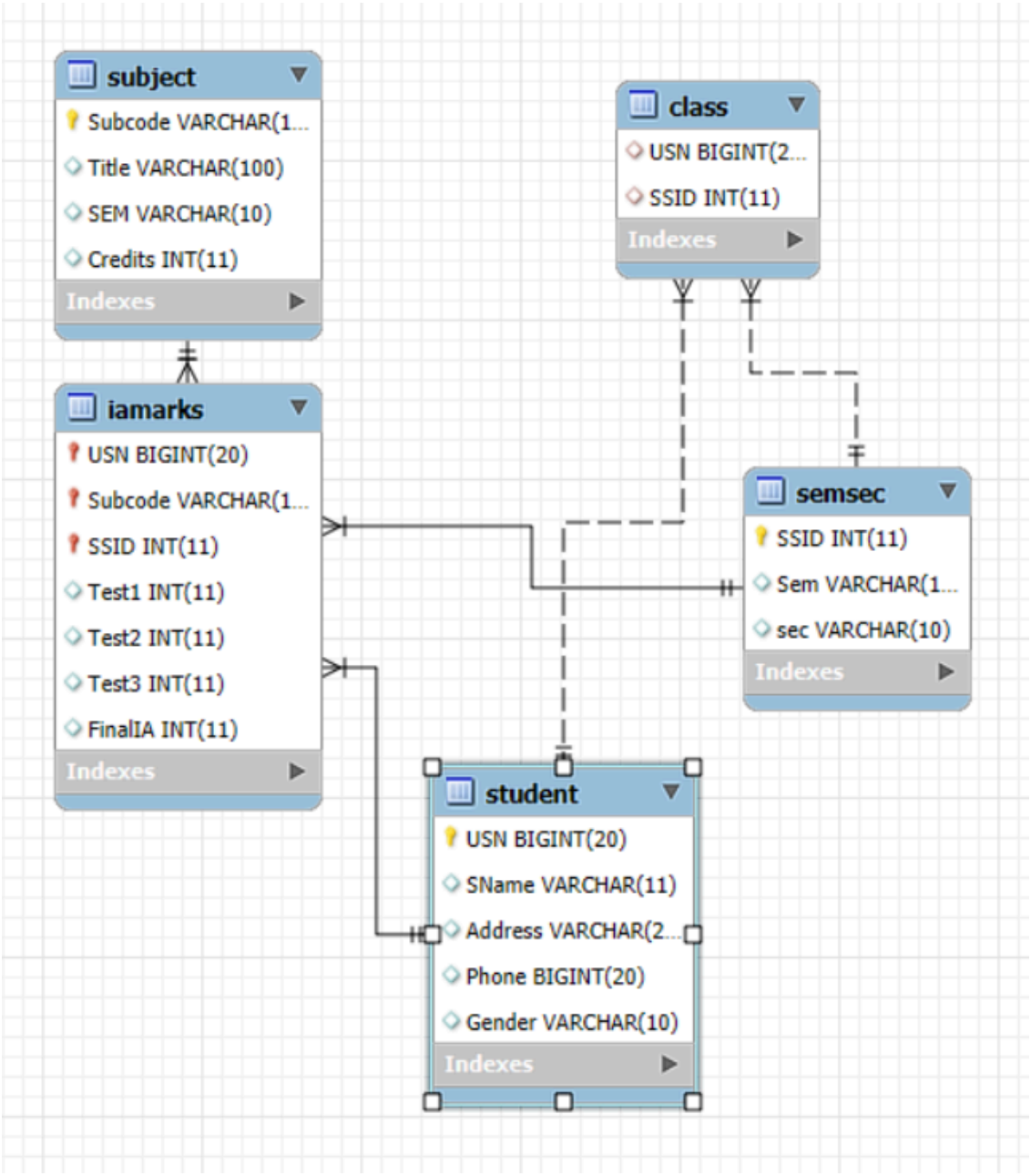


Table Descriptions and Dataset

STUDENT

- USN (BIGINT, Primary Key)
- SName (VARCHAR(11))
- Address (VARCHAR(20))
- Phone (BIGINT)
- Gender (VARCHAR(10))

Student Sample Data:

USN	SName	Address	Phone	Gender
101	Alia	Bangalore	9876543210	F
102	Ranveer	Mysore	123456789	M

SEMSEC

- SSID (INT, Primary Key)
- Sem (VARCHAR(10))
- Sec (VARCHAR(10))

EMSEC Sample Data:

SSID	Sem	Sec
1	4th	C
2	4th	A

CLASS

- USN (BIGINT, Foreign Key)
- SSID (INT, Foreign Key)

Class Sample Data:

USN	SSID
101	5
102	1

SUBJECT

- Subcode (VARCHAR(10), Primary Key)
- Title (VARCHAR(100))
- Sem (VARCHAR(10))
- Credits (INT)

Subject Sample Data:

Subcode	Title	Sem	Credits
CA401	Database Management	4th	4
CA402	Operating Systems	4th	3

IAMARKS

- USN (BIGINT, Foreign Key)
- Subcode (VARCHAR(10), Foreign Key)
- SSID (INT, Foreign Key)
- Test1 (INT)
- Test2 (INT)
- Test3 (INT)
- FinalIA (INT)

IAMarks Sample Data:

USN	Subcode	SSID	Test1	Test2	Test3	FunalIA
101	CA401	5	12	14	10	26
102	CA402	1	15	18	20	38

Query 1. List all the student details studying in the fourth semester ‘C’ section.

```
select * from STUDENT S
inner join CLASS C on S.USN=C.USN
inner join SEMSEC SS on C.SSID =SS.SSID
where SS.Sem='4th'and SS.Sec='C';
```

	USN	SName	Address	Phone	Gender	USN	SSID	SSID	Sem	sec
▶	102	Ranvir	Mysore	9876543211	M	102	1	1	4th	C

Query 2. Compute the total number of male and female students in each semester and in each section.

```
SELECT SS.Sem, SS.Sec, S.Gender, S.USN, COUNT(*) AS Total_Students
FROM STUDENT S
INNER JOIN CLASS C ON S.USN = C.USN
INNER JOIN SEMSEC SS ON C.SSID = SS.SSID
GROUP BY SS.Sem, SS.Sec, S.Gender;
```

Sem	Sec	Gender	USN	Total_Students
4th	A	M	104	1
4th	C	M	102	1
6th	A	F	101	1
8th	B	F	105	1
8th	C	F	103	1

Query 3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE OR REPLACE VIEW Test1_Marks_View AS
SELECT I.USN, I.Subcode, S.Title, I.Test1
FROM IAMARKS I
INNER JOIN SUBJECT S ON I.Subcode = S.Subcode
WHERE I.USN = 101;
SELECT * FROM Test1_Marks_View;
```

USN	Subcode	Title	Test1
101	CA401	Database Management	12

Query 4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.

```
SET SQL_SAFE_UPDATES = 0;

UPDATE IAMARKS

SET FinalIA = ( (Test1 + Test2 + Test3) - LEAST(Test1, Test2, Test3) ) / 2;

SET SQL_SAFE_UPDATES = 1;

SELECT USN, Subcode, SSID, Test1, Test2, Test3, FinalIA FROM IAMARKS;
```

USN	Subcode	SSID	Test1	Test2	Test3	FinalIA
101	CA401	5	12	14	10	26
102	CA402	1	15	18	20	38
103	CA403	4	18	17	16	35
104	CA405	2	20	15	10	35
105	CA406	3	10	11	10	21

Query 5. Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding'

If FinalIA = 12 to 16 then CAT = 'Average'

If FinalIA< 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C section students.

```
SELECT S.USN, S.SName, SS.Sem, SS.Sec, I.FinalIA,
    CASE
        WHEN I.FinalIA BETWEEN 35 AND 40 THEN 'Outstanding'
        WHEN I.FinalIA BETWEEN 25 AND 30 THEN 'Average'
        WHEN I.FinalIA < 25 THEN 'Weak'
        ELSE 'Uncategorized'
    END AS CAT
FROM STUDENT S
INNER JOIN CLASS C ON S.USN = C.USN
INNER JOIN SEMSEC SS ON C.SSID = SS.SSID
INNER JOIN IAMARKS I ON S.USN = I.USN
WHERE SS.Sem = '8th' AND SS.Sec IN ('A', 'B', 'C');
```

USN	SName	Sem	Sec	FinalIA	CAT
103	Deepika	8th	C	35	Outstanding
105	Urvashi	8th	B	21	Weak