UNIVERSITY MANAGEMENT SYSTEM

DATABASE MANAGEMENT SYSTEM PROJECT REPORT

By

KOSIKA DHEERAJ REDDY

21MEB0A35

K. NAGA RITVIK

21EEB0B26

PROBLEM STATEMENT:

A university wants to create a database system to manage its operations effectively. The system should track information about subjects, professors, students, books, and departments. The following problem statement outlines the requirements for the database system:

Departments: The university has multiple departments identified by a unique department ID. Each department has a name, a head of department (HOD) name, and contact information (HOD phone number).

Students: Students are registered in the university and have access to the library. Each student has a unique student ID and is identified by their name, email, phone number, date of birth, and gender. Students are associated with a specific department.

Books: The library maintains a collection of books that can be borrowed by students. Each book has a unique book ID, a name, and a date of borrowing. The book is associated with the student who borrowed it.

Subjects: The department offers various subjects, each identified by a unique subject ID. Each subject has a name and is associated with a department.

Professors: Professors teach subjects at the university. Each professor has a unique professor ID and is associated with a subject, department, and contact information (phone number). Professors are identified by their names.

The database system should allow the following operations:

Add new subjects, professors, students, books, and departments to their respective tables.

Update information for existing subjects, professors, students, books, and departments.

Retrieve information about subjects, professors, students, books, and departments.

Delete subjects, professors, students, books, and departments from the database if necessary.

The database system should enforce the following relationships:

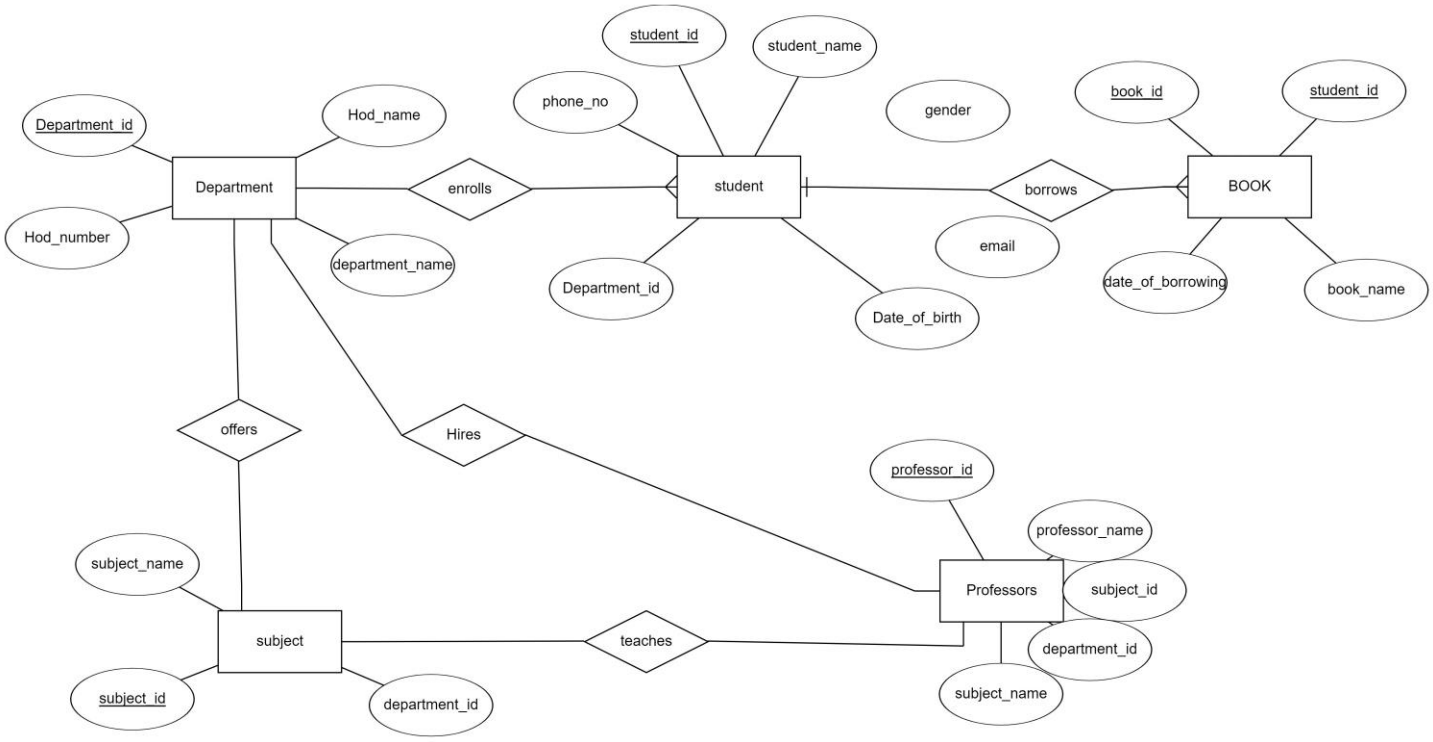
Professors are associated with a subject and a department. The subject ID and department ID in the PROFESSOR table should reference the corresponding IDs in the SUBJECTS and DEPARTMENT tables, respectively.

Books are associated with a student. The student ID in the BOOKS table should reference the corresponding ID in the STUDENT table.

Students are associated with a department. The department ID in the STUDENT table should reference the corresponding ID in the DEPARTMENT table.

The university management system aims to provide efficient tracking and management of subjects, professors, students, books, and departments. By maintaining this database, the library can easily manage borrowing records, professor assignments, student information, and departmental details, ensuring smooth operations within the university library environment.

ER DIAGRAM:



TABLES:

1)DEPARTMENT TABLE:

Field	Type	Null	Key	Default	Extra
department_id	varchar(3)	NO	PRI	NULL	
department_name	varchar(100)	NO		NULL	
Hod_name	varchar(100)	NO		NULL	
Hod_number	int	YES		NULL	

2)STUDENT TABLE

Field	Type	Null	Key	Default	Extra
student_id	int	NO	PRI	NULL	
student_name	varchar(100)	YES		NULL	
email	varchar(100)	NO		NULL	
date_of_birth	date	NO		NULL	
gender	varchar(100)	NO		NULL	
phone_number	int	NO		NULL	
department_id	varchar(3)	YES	MUL	NULL	

3)BOOKS TABLE:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

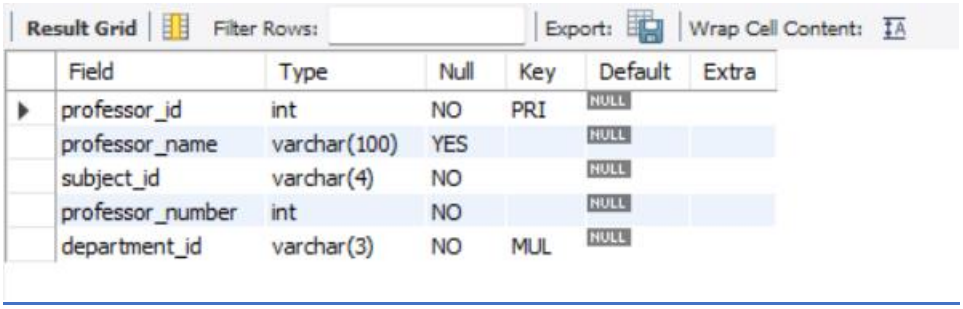
Field	Type	Null	Key	Default	Extra
book_id	int	NO	PRI	NULL	
book_name	varchar(100)	NO		NULL	
date_of_borrowing	date	NO		NULL	
student_id	int	YES	MUL	NULL	

4)SUBJECTS TABLE:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
subject_id	varchar(4)	NO	PRI	NULL	
subject_name	varchar(100)	NO		NULL	
department_id	varchar(3)	YES	MUL	NULL	

5)PROFESSOR TABLE:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
professor_id	int	NO	PRI	NULL	
professor_name	varchar(100)	YES		NULL	
subject_id	varchar(4)	NO		NULL	
professor_number	int	NO		NULL	
department_id	varchar(3)	NO	MUL	NULL	

FUNCTIONAL DEPENDENCIES AND PRIMARY KEYS:

- In the STUDENT table, we have the following functional dependencies:
  STUDENT_ID -> EMAIL, PHONE_NUMBER, STUDENT_NAME, DATE_OF_BIRTH, GENDER.
- In the BOOKS table, we have the following functional dependencies:
  BOOK_ID -> BOOK_NAME, DATE_OF_BORROWING STUDENT_ID -> [no functional dependencies]
- In the DEPARTMENT table, we have the following functional dependencies:
  DEPARTMENT_ID -> DEPARTMENT_NAME, HOD_PHONE_NO, HOD_NAME.
- In the PROFESSOR table, we have the following functional dependencies:
  PROFESSOR_ID -> SUBJECT_ID, PHONE_NUMBER, NAME, DEPARTMENT_ID
- In the SUBJECTS table, we have the following functional dependencies:
  SUBJECT_ID -> SUBJECT_NAME, DEPARTMENT_ID

ASSUMPTIONS:

1. Every student must enroll in a department and can enroll in only one department.
2. Students enrolled in a department must study all the subjects offered by the particular department.
3. Students can take more than one book at a time.

NORMALIZATION INTO HIGHER FORMS:

The three normal forms commonly referred to in database normalization are:

First Normal Form (1NF):

The first normal form requires that each column in a table contain only atomic values, meaning that each value should be indivisible. It eliminates repeating groups and ensures that each column has a single value. In 1NF, a table should have a primary key that uniquely identifies each row.

Second Normal Form (2NF):

The second normal form builds upon the first normal form. It states that a table should meet 1NF requirements and that all non-key attributes (columns) should be fully functionally dependent on the entire primary key. In other words, if a table has a composite primary key (multiple columns), each non-key column should depend on the entire composite key, not just a part of it. If a non-key column depends on only a portion of the primary key, it should be moved to a separate table.

Third Normal Form (3NF):

The third normal form goes further by eliminating transitive dependencies. It states that a table should meet the requirements of 2NF and that no non-key attribute should depend on another non-key attribute. In simpler terms, all non-key columns should be functionally dependent only on the primary key. If a non-key column depends on another non-key column, it should be moved to a separate table.

By following the normalization process and achieving higher normal forms, databases can minimize redundancy, improve data integrity, and simplify data maintenance and updates. Our table is already in Normalized form.

QUERIES FOR TABLE CREATION:

1.

```
create table department(
department_id VARCHAR(3) PRIMARY KEY,
department_name VARCHAR(100) NOT NULL,
Hod_name VARCHAR(100) NOT NULL,
Hod_number INTEGER
)
```

2.

```
create table student(
student_id integer PRIMARY KEY,
student_name VARCHAR(100),
email VARCHAR(100) NOT NULL,
date_of_birth date NOT NULL,
gender VARCHAR(100) NOT NULL,
phone_number INTEGER NOT NULL,
department_id VARCHAR(3),
foreign key(department_id) references department(department_id) on delete set Null
)
```

3.

```
create table books(
book_id integer PRIMARY KEY,
book_name VARCHAR(100) NOT NULL,
date_of_borrowing date NOT NULL,
student_id int,
foreign key(student_id) references student(student_id) on delete set Null
)
```

4.

```
create table subjects(
subject_id varchar(4) PRIMARY KEY,
subject_name VARCHAR(100) NOT NULL,
department_id VARCHAR(3),
foreign key(department_id) references department(department_id) on delete set Null
)
```

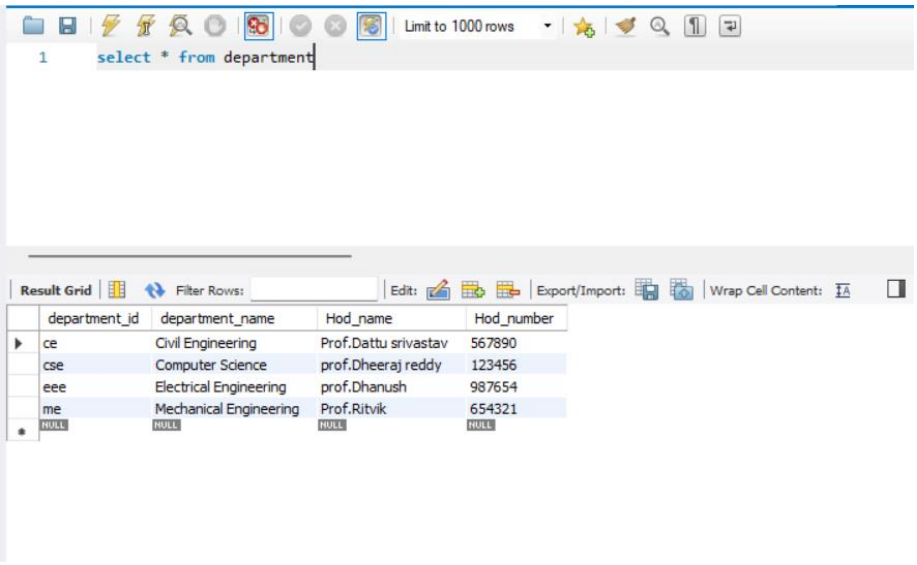
5.

```
create table professor(

professor_id integer PRIMARY KEY,

professor_name VARCHAR(50),

subject_id varchar(4) NOT NULL,
professor_number INTEGER NOT NULL,
department_id VARCHAR(3) NOT NULL,
foreign key(department_id) references department(department_id) on delete set Null
)
```

INSERTING RANDOM DATA:

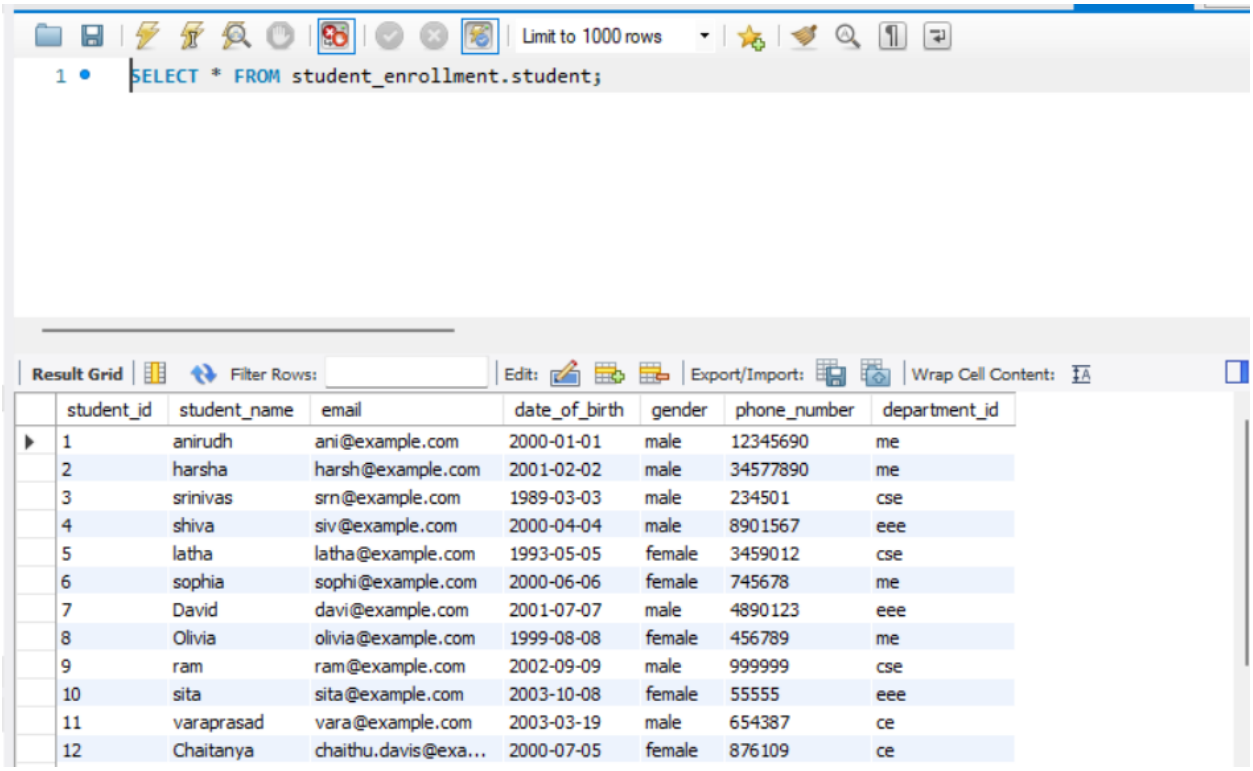
1. INSERTING DEPARTMENT TABLE

```
INSERT INTO department (department_id, department_name, Hod_name, Hod_number)

VALUES
    ('cse', 'Computer Science', 'prof.Dheeraj reddy', 123456),
    ('eee', 'Electrical Engineering','prof.Dhanush ', 987654),
    ('me', 'Mechanical Engineering', 'Prof.Ritvik', 654321),
    ('ce', 'Civil Engineering', 'Prof.Dattu srivastav', 567890);
```



2. INSERTING STUDENT TABLE

INSERT INTO student (student_id, student_name, email, date_of_birth, gender, phone_number, department_id)
VALUES
(1, 'anirudh', 'ani@example.com', '2000-01-01', 'male', 12345690, 'me'),
(2, 'harsha', 'harsh@example.com', '2001-02-02', 'male', 34577890, 'me'),
(3, 'srinivas', 'srn@example.com', '1989-03-03', 'male', 234501, 'cse'),
(4, 'shiva', 'siv@example.com', '2000-04-04', 'male', 8901567, 'eee'),
(5, 'latha', 'latha@example.com', '1993-05-05', 'female', 3459012, 'cse'),
(6, 'sophia', 'sophi@example.com', '2000-06-06', 'female', 745678, 'me'),
(7, 'David', 'davi@example.com', '2001-07-07', 'male', 4890123, 'eee'),
(8, 'Olivia', 'olivia@example.com', '1999-08-08', 'female', 0456789, 'me'),
(9, 'ram', 'ram@example.com', '2002-09-09', 'male', 999999, 'cse'),
(10, 'sita', 'sita@example.com', '2003-10-08', 'female', 55555, 'eee'),
(11, 'varaprasad', 'vara@example.com', '2003-03-19', 'male', 654387, 'ce'),
(12, 'Chaitanya', 'chaithu.davis@example.com', '2000-07-05', 'female', 876109, 'ce'),
(13, 'williamson', 'willi@example.com', '2002-01-16', 'male', 987510, 'eee'),
(14, 'misty', 'misty@example.com', '2001-11-22', 'female', 543876, 'me'),
(15, 'steven', 'smith@example.com', '2003-05-08', 'male', 2109876543, 'ce');



```
SELECT * FROM student_enrollment.student;
```

Limit to 1000 rows

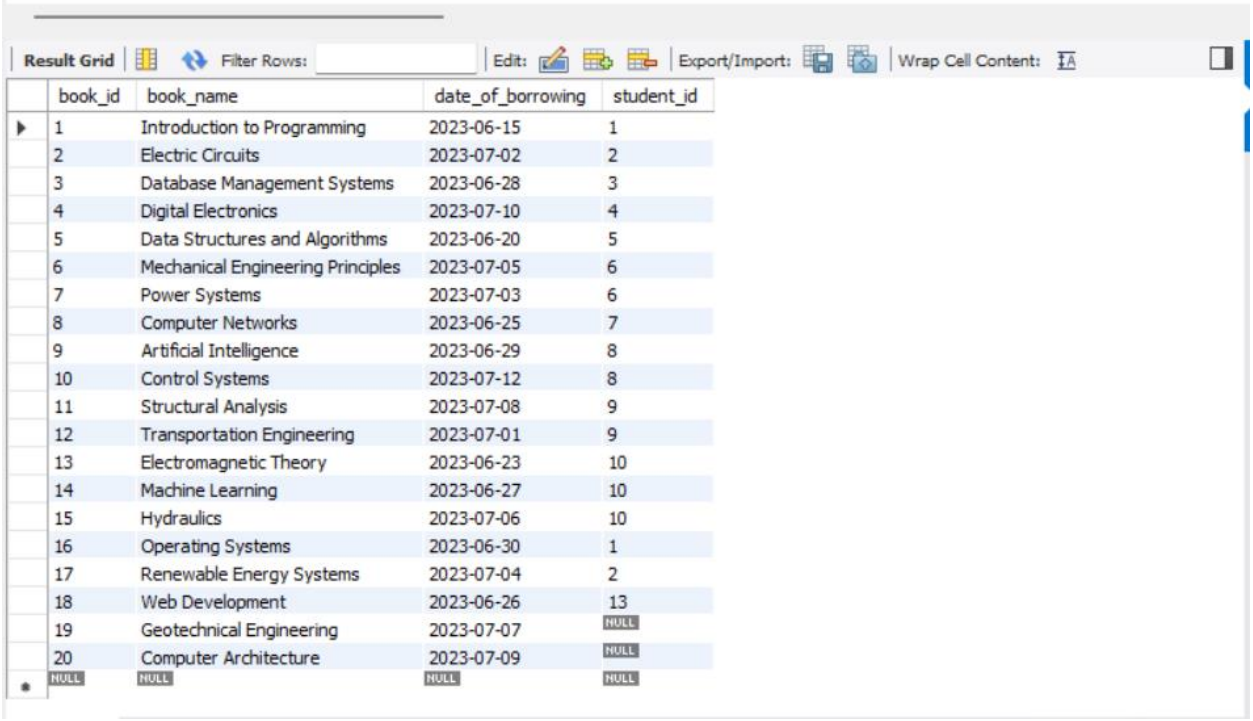
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

student_id	student_name	email	date_of_birth	gender	phone_number	department_id
1	anirudh	ani@example.com	2000-01-01	male	12345690	me
2	harsha	harsh@example.com	2001-02-02	male	34577890	me
3	srinivas	srn@example.com	1989-03-03	male	234501	cse
4	shiva	siv@example.com	2000-04-04	male	8901567	eee
5	latha	latha@example.com	1993-05-05	female	3459012	cse
6	sophia	sophi@example.com	2000-06-06	female	745678	me
7	David	davi@example.com	2001-07-07	male	4890123	eee
8	Olivia	olivia@example.com	1999-08-08	female	456789	me
9	ram	ram@example.com	2002-09-09	male	999999	cse
10	sita	sita@example.com	2003-10-08	female	55555	eee
11	varaprasad	vara@example.com	2003-03-19	male	654387	ce
12	Chaitanya	chaithu.davis@exa...	2000-07-05	female	876109	ce

3. INSERTING BOOK TABLE

```
INSERT INTO books (book_id, book_name, date_of_borrowing, student_id)
VALUES
    (1, 'Introduction to Programming', '2023-06-15', 1),
    (2, 'Electric Circuits', '2023-07-02', 2),
    (3, 'Database Management Systems', '2023-06-28', 3),
    (4, 'Digital Electronics', '2023-07-10', 4),
    (5, 'Data Structures and Algorithms', '2023-06-20', 5),
    (6, 'Mechanical Engineering Principles', '2023-07-05', 6),
    (7, 'Power Systems', '2023-07-03', 6),
    (8, 'Computer Networks', '2023-06-25', 7),
    (9, 'Artificial Intelligence', '2023-06-29', 8),
    (10, 'Control Systems', '2023-07-12', 8),
    (11, 'Structural Analysis', '2023-07-08', 9),
    (12, 'Transportation Engineering', '2023-07-01', 9),
    (13, 'Electromagnetic Theory', '2023-06-23', 10),
    (14, 'Machine Learning', '2023-06-27', 10),
    (15, 'Hydraulics', '2023-07-06', 10),
    (16, 'Operating Systems', '2023-06-30', 1),
    (17, 'Renewable Energy Systems', '2023-07-04', 2),
    (18, 'Web Development', '2023-06-26', 13),
    (19, 'Geotechnical Engineering', '2023-07-07', NULL),
    (20, 'Computer Architecture', '2023-07-09', NULL);
```

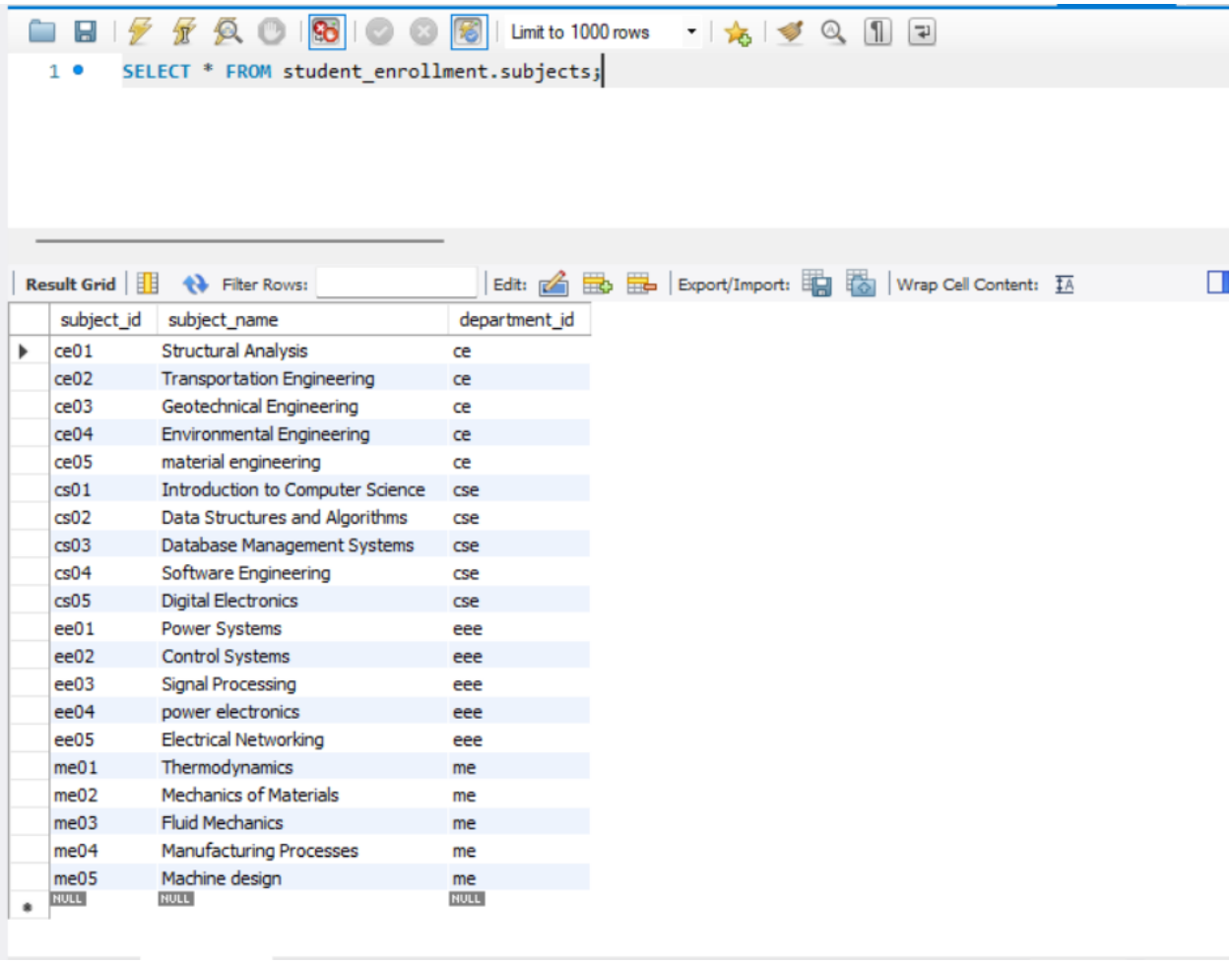
```
SELECT * FROM student_enrollment.books;
```



book_id	book_name	date_of_borrowing	student_id
1	Introduction to Programming	2023-06-15	1
2	Electric Circuits	2023-07-02	2
3	Database Management Systems	2023-06-28	3
4	Digital Electronics	2023-07-10	4
5	Data Structures and Algorithms	2023-06-20	5
6	Mechanical Engineering Principles	2023-07-05	6
7	Power Systems	2023-07-03	6
8	Computer Networks	2023-06-25	7
9	Artificial Intelligence	2023-06-29	8
10	Control Systems	2023-07-12	8
11	Structural Analysis	2023-07-08	9
12	Transportation Engineering	2023-07-01	9
13	Electromagnetic Theory	2023-06-23	10
14	Machine Learning	2023-06-27	10
15	Hydraulics	2023-07-06	10
16	Operating Systems	2023-06-30	1
17	Renewable Energy Systems	2023-07-04	2
18	Web Development	2023-06-26	13
19	Geotechnical Engineering	2023-07-07	NULL
20	Computer Architecture	2023-07-09	NULL
NULL	NULL	NULL	NULL

4. INSERTING SUBJECT TABLE

```
INSERT INTO subjects (subject_id, subject_name, department_id)
VALUES
    ('cs01', 'Introduction to Computer Science', 'cse'),
    ('cs02', 'Data Structures and Algorithms', 'cse'),
    ('cs03', 'Database Management Systems', 'cse'),
    ('cs04', 'Software Engineering', 'cse'),
    ('cs05', 'Digital Electronics', 'cse'),
    ('ee01', 'Power Systems', 'eee'),
    ('ee02', 'Control Systems', 'eee'),
    ('ee03', 'Signal Processing', 'eee'),
    ('me01', 'Thermodynamics', 'me'),
    ('me02', 'Mechanics of Materials', 'me'),
    ('me03', 'Fluid Mechanics', 'me'),
    ('me04', 'Manufacturing Processes', 'me'),
    ('ce01', 'Structural Analysis', 'ce'),
    ('ce02', 'Transportation Engineering', 'ce'),
    ('ce03', 'Geotechnical Engineering', 'ce'),
    ('ce04', 'Environmental Engineering', 'ce'),
    ('ee04', 'power electronics', 'eee'),
    ('me05', 'Machine design', 'me'),
    ('ee05', 'Electrical Networking', 'eee'),
    ('ce05', 'material engineering', 'ce');
```

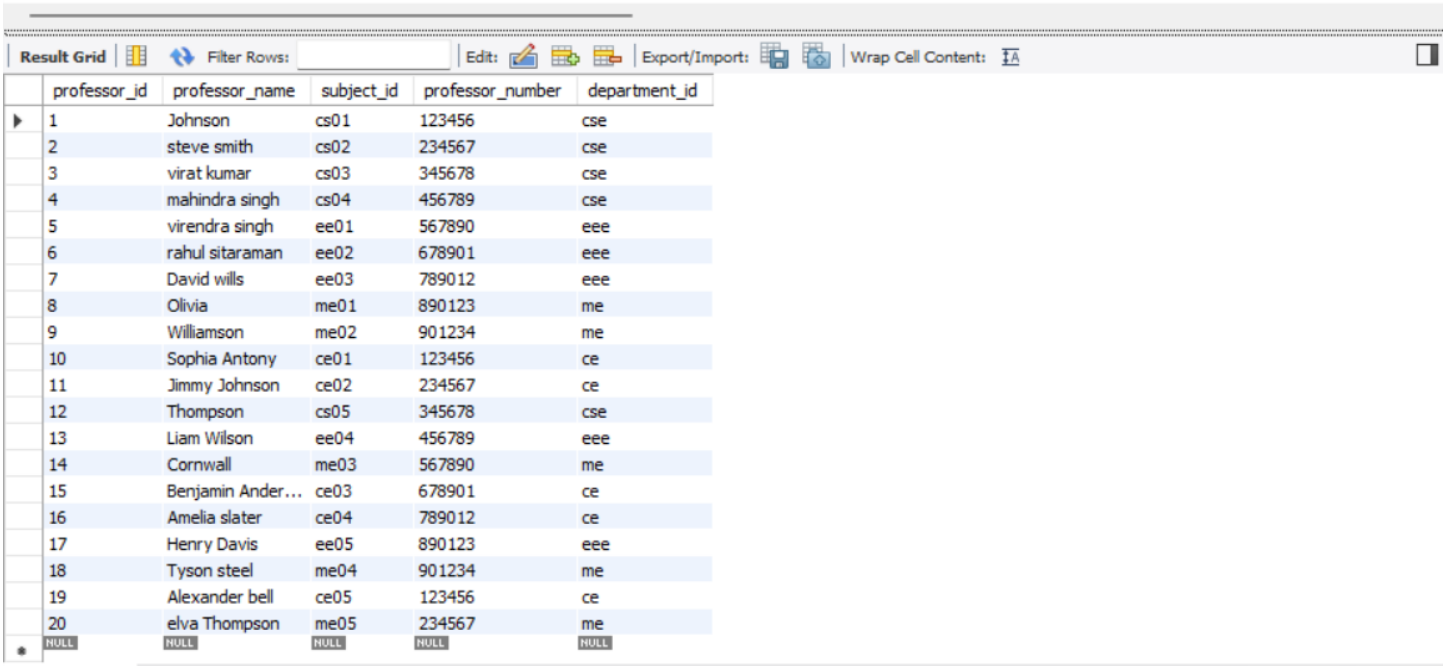


subject_id	subject_name	department_id
ce01	Structural Analysis	ce
ce02	Transportation Engineering	ce
ce03	Geotechnical Engineering	ce
ce04	Environmental Engineering	ce
ce05	material engineering	ce
cs01	Introduction to Computer Science	cse
cs02	Data Structures and Algorithms	cse
cs03	Database Management Systems	cse
cs04	Software Engineering	cse
cs05	Digital Electronics	cse
ee01	Power Systems	eee
ee02	Control Systems	eee
ee03	Signal Processing	eee
ee04	power electronics	eee
ee05	Electrical Networking	eee
me01	Thermodynamics	me
me02	Mechanics of Materials	me
me03	Fluid Mechanics	me
me04	Manufacturing Processes	me
me05	Machine design	me
NULL	NULL	NULL

1. INSERTING PROFESSOR TABLE

```
INSERT INTO professor (professor_id, professor_name, subject_id, professor_number, department_id)
VALUES
    (1, 'Johnson', 'cs01', 123456, 'cse'),
    (2, 'steve smith', 'cs02', 234567, 'cse'),
    (3, 'virat kumar', 'cs03', 345678, 'cse'),
    (4, 'mahindra singh', 'cs04', 456789, 'cse'),
    (5, 'virendra singh', 'ee01', 567890, 'eee'),
    (6, 'rahul sitaraman', 'ee02', 678901, 'eee'),
    (7, 'David wills', 'ee03', 789012, 'eee'),
    (8, 'Olivia ', 'me01', 890123, 'me'),
    (9, 'Williamson', 'me02', 901234, 'me'),
    (10, 'Sophia Antony', 'ce01', 123456, 'ce'),
    (11, 'Jimmy Johnson', 'ce02', 234567, 'ce'),
    (12, 'Thompson', 'cs05', 345678, 'cse'),
    (13, 'Liam Wilson', 'ee04', 456789, 'eee'),
    (14, 'Cornwall', 'me03', 567890, 'me'),
    (15, 'Benjamin Anderson', 'ce03', 678901, 'ce'),
    (16, 'Amelia slater', 'ce04', 789012, 'ce'),
    (17, 'Henry Davis', 'ee05', 890123, 'eee'),
    (18, 'Tyson steel', 'me04', 901234, 'me'),
    (19, 'Alexander bell', 'ce05', 123456, 'ce'),
    (20, 'elva Thompson', 'me05', 234567, 'me');
```

```
1    select * from professor
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

professor_id	professor_name	subject_id	professor_number	department_id
1	Johnson	cs01	123456	cse
2	steve smith	cs02	234567	cse
3	virat kumar	cs03	345678	cse
4	mahindra singh	cs04	456789	cse
5	virendra singh	ee01	567890	eee
6	rahul sitaraman	ee02	678901	eee
7	David wills	ee03	789012	eee
8	Olivia	me01	890123	me
9	Williamson	me02	901234	me
10	Sophia Antony	ce01	123456	ce
11	Jimmy Johnson	ce02	234567	ce
12	Thompson	cs05	345678	cse
13	Liam Wilson	ee04	456789	eee
14	Cornwall	me03	567890	me
15	Benjamin Ander...	ce03	678901	ce
16	Amelia slater	ce04	789012	ce
17	Henry Davis	ee05	890123	eee
18	Tyson steel	me04	901234	me
19	Alexander bell	ce05	123456	ce
20	elva Thompson	me05	234567	me
NULL	NULL	NULL	NULL	NULL

SQL QUERIES:

1. ### QUERY TO RETURN INFORMATION OF PROFESSORS IN A DEPARTMENT:

```
select * from professor
where professor.department_id='me';
```

2.

```
select * from professor
where professor.department_id='me';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

professor_id	professor_name	subject_id	professor_number	department_id
8	Olivia	me01	890123	me
9	Williamson	me02	901234	me
14	Cornwall	me03	567890	me
18	Tyson steel	me04	901234	me
20	elva Thompson	me05	234567	me
NULL	NULL	NULL	NULL	NULL

2.QUERY TO RETURN SUBJECTS OFFERED AND PROFESSORS TEACHING THE COURSE:

```
select subjects.subject_id,subjects.subject_name,professor.professor_id,
professor.professor_name,professor.department_id
from subjects
join professor on professor.subject_id=subjects.subject_id
order by subject_id asc;
```

```
select subjects.subject_id,subjects.subject_name,professor.professor_id,
professor.professor_name,professor.department_id
from subjects
join professor on professor.subject_id=subjects.subject_id
order by subject_id asc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

subject_id	subject_name	professor_id	professor_name	department_id
ce01	Structural Analysis	10	Sophia Antony	ce
ce02	Transportation Engineering	11	Jimmy Johnson	ce
ce03	Geotechnical Engineering	15	Benjamin Anderson	ce
ce04	Environmental Engineering	16	Amelia slater	ce
ce05	material engineering	19	Alexander bell	ce
cs01	Introduction to Computer Science	1	Johnson	cse
cs02	Data Structures and Algorithms	2	steve smith	cse
cs03	Database Management Systems	3	virat kumar	cse
cs04	Software Engineering	4	mahindra singh	cse
cs05	Digital Electronics	12	Thompson	cse
ee01	Power Systems	5	virendra singh	eee
ee02	Control Systems	6	rahul sitaraman	eee
ee03	Signal Processing	7	David wills	eee
ee04	power electronics	13	Liam Wilson	eee
ee05	Electrical Networking	17	Henry Davis	eee
me01	Thermodynamics	8	Olivia	me
me02	Mechanics of Materials	9	Williamson	me
me03	Fluid Mechanics	14	Cornwall	me
me04	Manufacturing Processes	18	Tyson steel	me
me05	Machine design	20	elva Thompson	me

3. query to retrieves the information related to a specific student like name, the subjects they are enrolled in, the corresponding professors for those subjects, the department name, and the name of the Head of the Department (HOD).

```
Select
student.student_name,subjects.subject_id,subjects.subject_name,professor.professor_nam
e,
department.department_name,department.Hod_name
from student
left join subjects on subjects.department_id=student.department_id
left join professor on subjects.subject_id=professor.subject_id
left join department on department.department_id=professor.department_id
where student_id='3'
```

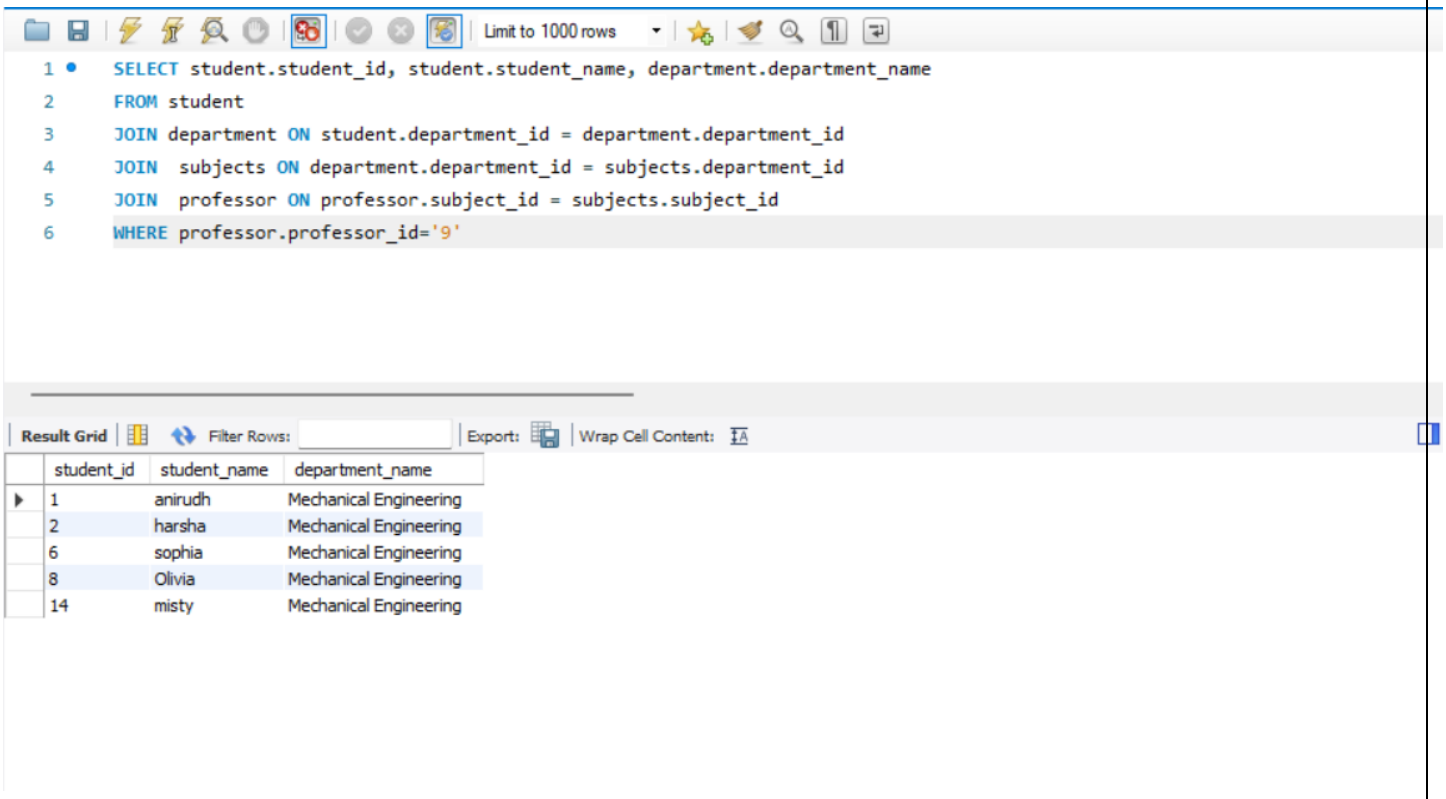
```
select student.student_name,subjects.subject_id,subjects.subject_name,professor.professor_name,
department.department_name,department.Hod_name
from student
left join subjects on subjects.department_id=student.department_id
left join professor on subjects.subject_id=professor.subject_id
left join department on department.department_id=professor.department_id
where student_id='3'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

student_name	subject_id	subject_name	professor_name	department_name	Hod_name
srinivas	cs01	Introduction to Computer Science	Johnson	Computer Science	prof.Dheeraj reddy
srinivas	cs02	Data Structures and Algorithms	steve smith	Computer Science	prof.Dheeraj reddy
srinivas	cs03	Database Management Systems	virat kumar	Computer Science	prof.Dheeraj reddy
srinivas	cs04	Software Engineering	mahindra singh	Computer Science	prof.Dheeraj reddy
srinivas	cs05	Digital Electronics	Thompson	Computer Science	prof.Dheeraj reddy

4. Query to return the list of students studying under a particular professor:

```
SELECT student.student_id, student.student_name,
department.department_name

FROM student

JOIN department ON student.department_id = department.department_id

JOIN  subjects ON department.department_id = subjects.department_id

JOIN  professor ON professor.subject_id = subjects.subject_id

WHERE professor.professor_id='9'
```



```
SELECT student.student_id, student.student_name, department.department_name
FROM student
JOIN department ON student.department_id = department.department_id
JOIN  subjects ON department.department_id = subjects.department_id
JOIN  professor ON professor.subject_id = subjects.subject_id
WHERE professor.professor_id='9'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

student_id	student_name	department_name
1	anirudh	Mechanical Engineering
2	harsha	Mechanical Engineering
6	sophia	Mechanical Engineering
8	Olivia	Mechanical Engineering
14	misty	Mechanical Engineering