STUDENT_DB
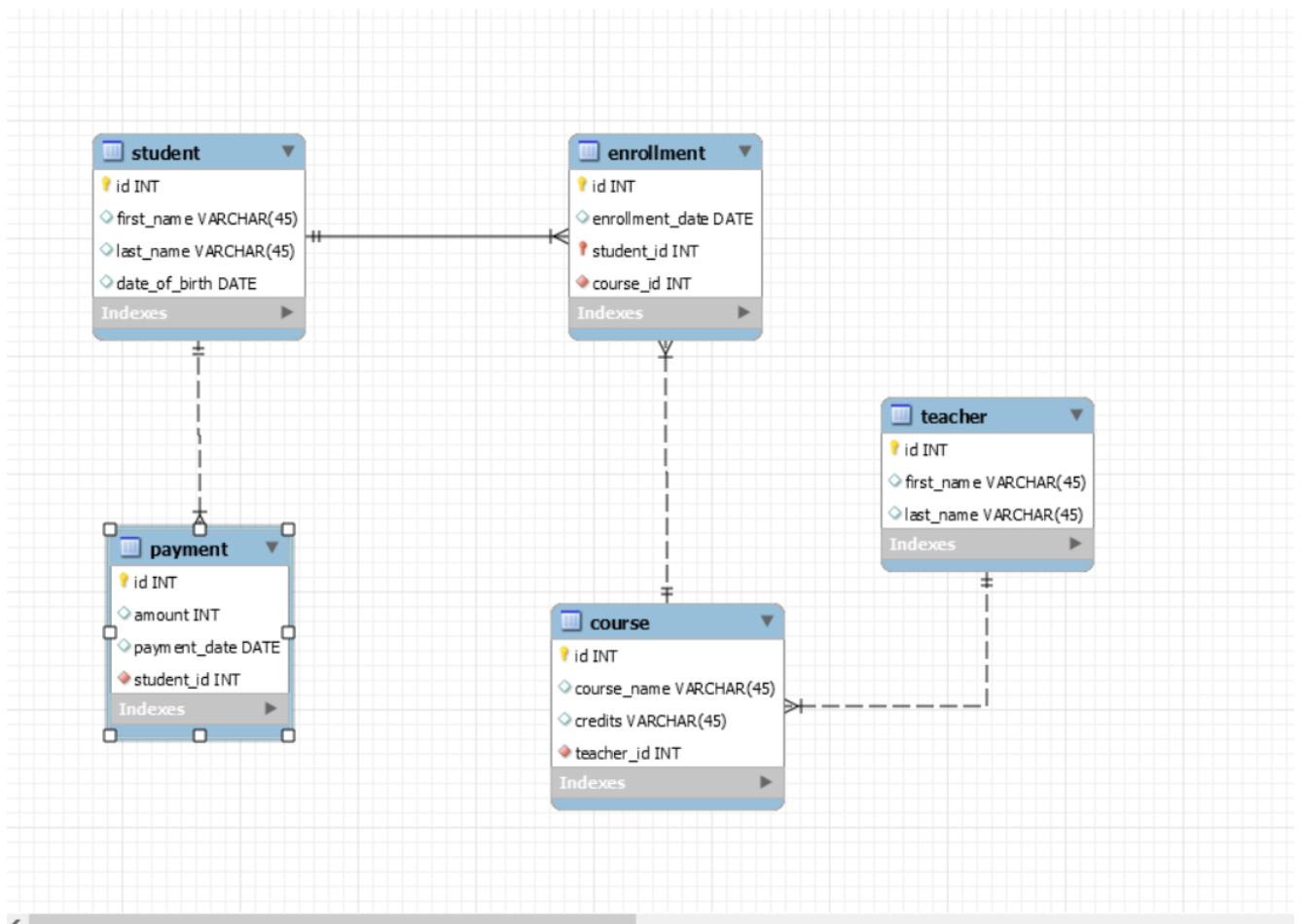

show databases;

use student_db;

show tables;

desc student;

/*insert into student(id,first_name,last_name,date_of_birth) values

(1,'nila','ram','2003-02-09'),

(2,'sheela','sam','2002-02-19'),

(3,'sanjay','durai','2002-12-29'),

(4,'kumar','stalin','2003-10-09'); desc teacher;

```sql
/*insert into teacher (id,first_name,last_name)values (1,'suresh','ramesh'),

(5,'alex','matt'),

(4,'kanth','bharathi'),

(3,'ravi','tharun'),

(2,'harish','dinesh');

insert into course(course_name,credits,teacher_id)values ('cse','40',1),

('Ece','30',5),

('Aids','50',4),

('Mech','60',3),

('IT','50',2);*/

desc enrollment;




/*insert into enrollment(enrollment_date,student_id,course_id) values ('2020-05-06',1,1),

('2022-05-06',3,2),


('2021-08-02',2,3),

('2021-04-06',4,4),

('2020-12-16',1,2);*/


/*task 2*/



/*2. Write an SQL query to enroll a student in a course. Choose an existing student and course and insert a record into the "Enrollments"

table with the enrollment date*/


insert into enrollment(enrollment_date,student_id,course_id) values ('2024-05-04',1,1);


/*3.    Update the email address of a specific teacher in the "Teacher" table.
```

Choose any teacher and modify their email address.*/

```sql
update teacher set first_name='deena' where id=1;
```

/*4.   Write an SQL query to delete a specific enrollment record from the "Enrollments" table. Select an enrollment

record based on the student and course.

*/

```sql
delete from enrollment where course_id=2;
```

/*5. Update the "Courses" table to assign a specific teacher to a course. Choose any course and teacher from the respective tables.*/

```sql
select * from course;
```

```sql
update course set teacher_id='2' where course_name='cse';
```

/*6. Delete a specific student from the "Students" table and remove all their enrollment records from the "Enrollments" table. Be sure to maintain referential integrity.*/

```sql
DELETE FROM student
WHERE id IN (SELECT id FROM enrollment WHERE student_id = 1);
```

/*7. Update the payment amount for a specific payment record in the "Payments" table. Choose any payment record and modify the payment amount.*/

```sql
update payment set payment='6233' where student_id=1;
```

/* task 3*/

/*Write an SQL query to calculate the total payments made by a specific student. You will need to join the "Payments" table with the "Students" table based on the student's ID.*/

```
select sum(p.amount),s.id from payment p JOIN student s ON s.id=p.student_id group by p.student_id;
```

/*2. Write an SQL query to retrieve a list of courses along with the count of students enrolled in each course. Use a JOIN operation between the "Courses" table and the "Enrollments" table.*/

```
select e.course_id,c.course_name from course c JOIN enrollment e On c.id=e.course_id group by e.course_id;
```

/*3. Write an SQL query to find the names of students who have not enrolled in any course. Use a LEFT JOIN between the "Students" table and the "Enrollments" table to identify students

without enrollments.*/

```
select s.first_name,s.last_name from student s where NOT exists (select e.student_id from enrollment e group by course_id);
```

/*4. Write an SQL query to retrieve the first name, last name of students, and the names of the courses they are enrolled in. Use JOIN operations between the "Students" table and the

"Enrollments" and "Courses" tables.*/

```
select s.first_name,s.last_name,c.id,c.course_name,e.enrollment_date from student s JOIN enrollment e
ON s.id=e.student_id JOIN course c ON e.course_id=c.id;
```

/*5. Create a query to list the names of teachers and the courses they are assigned to. Join the "Teacher" table with the "Courses" table.*/

select t.first_name,c.course_name from teacher t JOIN course c ON c.teacher_id=t.id group by c.course_name;

/*6. Retrieve a list of students and their enrollment dates for a specific course. You'll need to join the "Students" table with the "Enrollments" and "Courses" tables*/

select s.first_name,s.last_name,c.course_name from student s JOIN course c ON s.id = c.student_id group by c.course_name;

/*7. Find the names of students who have not made any payments. Use a LEFT JOIN between the "Students" table and the "Payments" table and filter for students with NULL payment records.

8. Write a query to identify courses that have no enrollments. You'll need to use a LEFT JOIN between the "Courses" table and the "Enrollments" table and filter for courses with NULL

enrollment records.*/

select s.first_name,s.last_name from student s where NOT exists (select 1 FROM payment p WHERE p.student_id = s.id);

/*8. Write a query to identify courses that have no enrollments. You'll need to use a LEFT JOIN between the "Courses" table and the "Enrollments" table and filter for courses with NULL

enrollment records.*/

select c.id,c.course_name from course c where NOT exists(select 1 from enrollment e where c.id=e.course_id);

/*9. Identify students who are enrolled in more than one course. Use a self-join on the "Enrollments" table to find students with multiple enrollment records.*/

SELECT student_id FROM enrollment

GROUP BY student_id

HAVING COUNT(course_id) > 1;

/*10. Find teachers who are not assigned to any courses. Use a LEFT JOIN between the "Teacher" table and the "Courses" table and filter for teachers with NULL course assignments

*/

select id from teacher where NOT exists(select teacher_id from course);

/* Task 4. Subquery and its type:

1. Write an SQL query to calculate the average number of students enrolled in each course. Use aggregate functions and subqueries to achieve this.*/

select avg(num_student) from (select count(student_id) as num_student from enrollment)

 group by course_id; /*doubt*/

/*2. Identify the student(s) who made the highest payment. Use a subquery to find the maximum payment amount and then retrieve the student(s) associated with that amount.*/

select student_id from payment where amount =(select max(amount) from payment)

/*3. Retrieve a list of courses with the highest number of enrollments. Use subqueries to find the course(s) with the maximum enrollment count.*/

select course_id from enrollment where (select count(student_id) from enrollment group by course_id

order by count(student_id) desc limit 1);

/*4. Calculate the total payments made to courses taught by each teacher. Use subqueries to sum payments for each teacher's courses.*/

select*from enrollment;

select sum(amount),id from payment where id in(select id from teacher) group by(student_id);

/*5. Identify students who are enrolled in all available courses. Use subqueries to compare a student's enrollments with the total number of courses.*/

select student_id from enrollment group by student_id having count(distinct course_id)=(select count(*) from course);

/* doubt*/

/*6. Retrieve the names of teachers who have not been assigned to any courses. Use subqueries to find teachers with no course assignments.*/

SELECT id, first_name FROM teacher

WHERE id not in

(SELECT teacher_id from course);

/*7. Calculate the average age of all students. Use subqueries to calculate the age of each student based on their date of birth.*/

/*select avg(age) from (select*/

/* doubt*/

/*8. Identify courses with no enrollments. Use subqueries to find courses without enrollment records.*/

select id from course where not exists (select course_id from enrollment

where enrollment.course_id=course.id) ;

/*9. Calculate the total payments made by each student for each course they are enrolled in. Use subqueries and aggregate functions to sum payments.*/

select student_id,sum(amount) from payment where student_id in(select student_id from enrollment)

group by student_id;

/*10. Identify students who have made more than one payment. Use subqueries and aggregate functions to count payments per student and filter for those with counts

greater than one.*/

select student_id,count(id),id from payment group by student_id having count(id)>1;

/*11. Write an SQL query to calculate the total payments made by each student. Join the "Students" table with the "Payments" table and use GROUP BY to calculate the sum of payments for each

student.*/

select sum(p.amount),s.first_name from student s join payment p on s.id=p.student_id group by p.student_id;

/*12.   Retrieve a list of course names along with the count of students enrolled in each course. Use JOIN operations between the "Courses" table and the

 "Enrollments" table and GROUP BY to count enrollments.*/

select c.id,c.course_name,e.enrollment_date,count(e.student_id) from course c JOIN enrollment e ON c.id=e.course_id group by course_id;

/*13.   Calculate the average payment amount made by students. Use JOIN operations between the "Students" table and

 the "Payments" table and GROUP BY to calculate the average */

```sql
select student_id,avg(amount) from payment where student_id in(select id from student) group by student_id;


/* no of student taught by each teacher;*/


select count(s.id),t.first_name from student s JOIN enrollment e ON e.student_id=s.id

JOIN course c ON e.course_id = c.id JOIN teacher t ON c.teacher_id = t.id

 group by t.first_name;
```