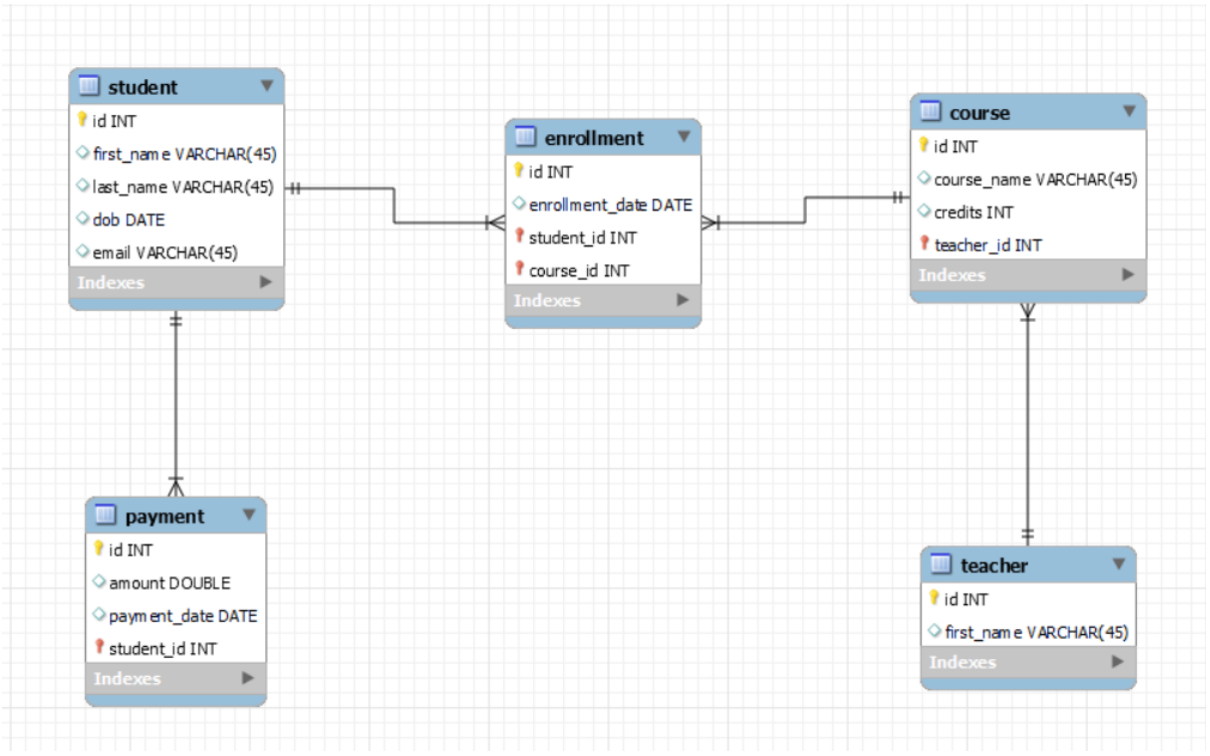
STUDENT INFORMATION SYSTEM



<u>CODE:</u>

```
use student_hex_feb_24;

#insertion

insert into student (first_name, last_name, dob, email) values
('diya','sara', '2002-02-04','diya@gmail.com'),
('dev', 'suriya','2002-03-10','dev@gmail.com'),
('tara','krish','2002-05-17', 'tara@gmail.com'),
('atul','bose' ,'2002-09-21','atul@gmail.com'),
('ajay','josh','2002-11-30', 'ajay@gmail.com');

insert into payment (amount, payment_date, student_id) values
(10000.0, '2024-03-04', 1),
(12000.0, '2024-02-28', 2),
(11000.0, '2024-03-06', 3),
```

```sql
(13000.0, '2024-03-05', 4),
(10000.0, '2024-03-03', 5);

insert into teacher (first_name) values
('misti'),
('abir'),
('kuhu'),
('kunal');

insert into course (course_name, credits, teacher_id) values
('c programming', 10, 3),
('java programming', 15, 2),
('sql', 9, 1);

insert into enrollment (enrollment_date, student_id, course_id) values
('2024-03-04', 1, 1),
('2024-02-28', 2,2),
('2024-03-06', 3,3),
('2024-03-05', 4,2),
('2024-03-03', 5,1);



# Task 2

-- 1. Write an SQL query to insert a new student into the "Students" table with the following details:
insert into student values
('john','doe','1995-08-15','john.doe@example.com',1234567890);

-- 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 values

('2024-03-07',5,2);

-- 3. Update the email address of a specific teacher in the "Teacher" table. Choose any teacher and modify their email address.

update teacher

set email='kunal@gmail.com' where id=4;

-- 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.

delete from enrollment

where student_id=3 AND course_id=3;

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

update course

set course_name='java fundamentals' where teacher_id=2;

-- 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.

delete from student

where id=6;

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

update payment

set amount=15000

where id=1;

# Task 3

-- 1. 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 s.id, p.id, SUM(p.amount) as payment

from payment p

JOIN student s ON s.id=p.id

where s.id=1;
```

-- 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 c.course_name, COUNT(e.student_id) as count

from course c

JOIN enrollment e ON e.course_id=c.id

group by c.course_name;
```

-- 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

left join enrollment e on s.id = e.student_id

where e.student_id IS NULL;
```

-- 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.course_name

from  student s

INNER 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
```

```sql
group by t.first_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, c.course_name, e.enrollment_date
from student s
INNER JOIN enrollment e ON s.id=e.student_id
JOIN course c ON e.course_id=c.id;

-- 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.
select s.id, s.first_name
from student s
INNER JOIN payment p ON s.id=p.student_id
where p.student_id IS NULL;

-- 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.course_name
from course c
INNER JOIN enrollment e ON e.course_id=c.id
where e.course_id IS NULL;

-- 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 t.first_name
from teacher t
LEFT JOIN course c on c.teacher_id=t.id
where c.teacher_id IS NULL;
```

Task 4

```
-- 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 *

from student

where id = (select student_id

                    from payment

                    order by amount desc

                    limit 0,1);
```

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

select teacher_id, sum(amount)

from payment p

JOIN enrollment e ON p.student_id = e.student_id

JOIN course c ON e.course_id = c.id

group by teacher_id;
```

```
-- 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 distinct teacher_id

                              from course);
```