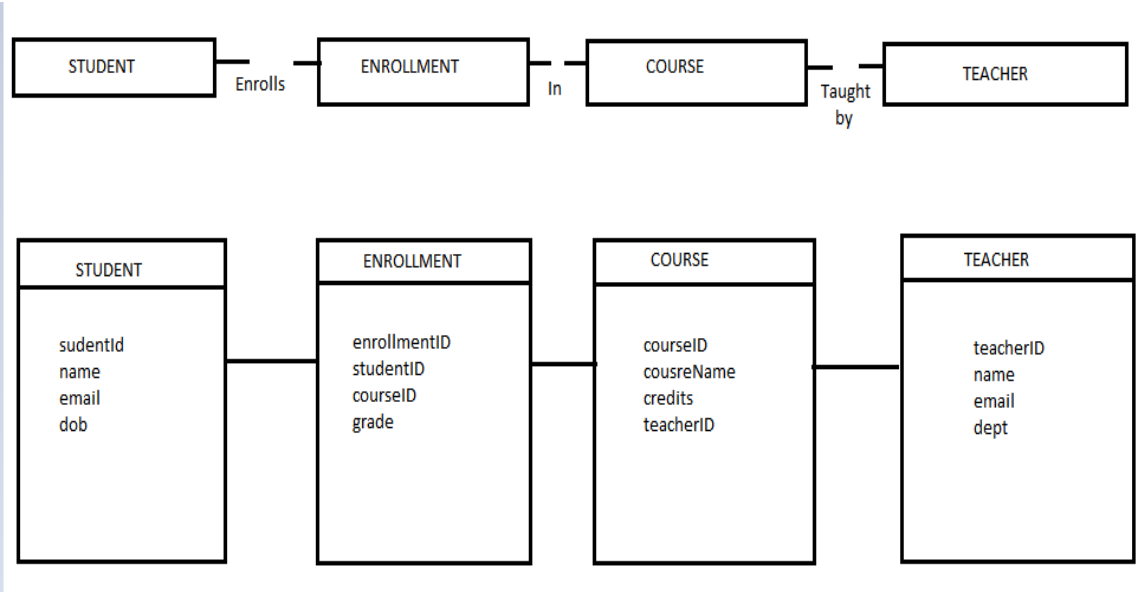
Assignment 1: Analyze a given business scenario and create an ER diagram that includes entities, relationships, attributes, and cardinality. Ensure that the diagram reflects proper normalization up to the third normal form.

ER Diagram:-



Entities:

Student, Teacher, Course, and Enrollment.

Attributes:

- ✓ · Student: studentID (Primary Key), name, email, dob
- ✓ · Teacher: teacherID (Primary Key), same, email, dept
- ✓ · Course: CourseID (Primary Key), courseName, credits, teacherID (Foreign Key)
- ✓ · Enrollment: enrollmentID (Primary Key),studentID (Foreign Key),courseID (Foreign Key), grade

Relationships:

- ✓ Students enroll in courses, and the Enrollment entity captures this many-to-many relationship.
- ✓ Courses are taught by teachers in a many-to-one relationship.

Cardinality:

- ✓ A student can enroll in many courses (1 to many), and each enrollment record links to one student (many to 1).
- ✓ A course can have many students (1 to many), and each enrollment record links to one course (many to 1).
- ✓ A teacher can teach many courses (1 to many), but each course is taught by one teacher (many to 1).

3 Normal Form (3NF):-

- ✓ **1NF**: All attributes are atomic (e.g., name, email, dob are single values, not composite).
- ✓ **2NF**: All non-key attributes are fully functionally dependent on the primary key. For instance, courseName, credits, and teacherID depend on courseID, and name and email depend on teacherID.
- ✓ **3NF**: There are no transitive dependencies. For example, in the Enrollment entity, grade depends directly on the primary key enrollmentID.

==========

Assignment 2: Design a database schema for a library system, including tables, fields, and constraints like NOT NULL, UNIQUE, and CHECK. Include primary and foreign keys to establish relationships between tables.

```
Create table Authors
(
    authorId number(10) primary key ,
    firstName varchar2(30) not null,
    lastName varchar2(30) not null
);

Create table Categories
 (
    categoryId number(10)    primary key ,
    categoryName varchar2(30)    unique    not null
);

Create table Books
 (
    bookId number(10)      primary key,
    title varchar2(30) not null,
    publishedYear varchar2(30) not null check ,
    authorId number(10) not null,
    categoryId number(10) not null,
    Foreign key (authorId) references Authors(authorId),
    Foreign key (categoryId) references Categories(categoryId)
);

Create table Members
(
    memberId number(10)    primary key ,
    firstName varchar2(30) not null,
    lastName varchar2(30) not null,
    email varchar2(30) unique not null,
    phone varchar2(30) not null,
    address varchar2(30) not null,
    membershipDate varchar2(30) not null
);
```

==========

Assignment 1: Write a SELECT query to retrieve all columns from a 'customers' table, and modify it to return only the customer name and email address for customers in a specific city.

```
Create table customers
(
CustomerId number(10),
CustomerName varchar2(30,
Email varchar2(30),
City varchar2(30),
);

Insert into customer values (1,'Jude','jude@jj.com', 'new york');
Insert into customer values (2,'Liam','liam@ll.com', 'Hawei');
Insert into customer values (3,'Alex',alex@aa.com','San Andres');

Select * from customer;

Select CustomerName, Email from customer where city = 'new york';
```

==========

Assignment 2: Craft a query using an INNER JOIN to combine 'orders' and 'customers' tables for customers in a specified region, and a LEFT JOIN to display all customers including those without orders.

Query 1: INNER JOIN for customers in a specified region ('North')

```
select customers.customerId, customers.customerName, customers.region,
orders.orderId, orders.orderDate from customers
INNER JOIN orders on customers.customerId = orders.customerId
where customers.region = 'North';
```

Query 2: LEFT JOIN to display all customers including those without orders

```
select customers.customerId, customers.customerName, customers.region,
orders.orderId, orders.orderDate from customers
LEFT JOIN orders on customers.customerId = orders.customerId
```

==========

Assignment 4: Write SQL statements to CREATE a new database and tables that reflect the library schema you designed earlier. Use ALTER statements to modify the table structures and DROP statements to remove a redundant table.

```
create table Book
(
    bookID number(10) primary key,
    title varchar2(30) not null,
    author varchar2(30) not null,
);

Create table Member
```

```
(
	memberID number(10)    Primary Key,
	name varchar2(30)    not null,
	email varchar(30) unique    not null
);


Create table Loan
(
	loanID number(10) Primary Key,
	bookID number(10) ,
	memberID number(10) not null,
	loanDate varchar2(30)    not null,
	dueDate varchar2(30) not null,
	returnDate varchar2(30),
	Foreign key (bookID) references Book(bookID),
	Foreign key (memberID) references Member(memberID)
);


alter table Book add column publicationYear number(10);

alter table Member add column mobileNo varchar2(30);

alter table Loan modify column returnDate varchar2(30) not null;

drop table if exixts Publisher;

==========================================================================
```