CSE 560

Data Models and Query Languages

Project 1

Tiny Piazza

Submitted By

Karan Manchandia

UBIT Name: karanman

Person # 50290755

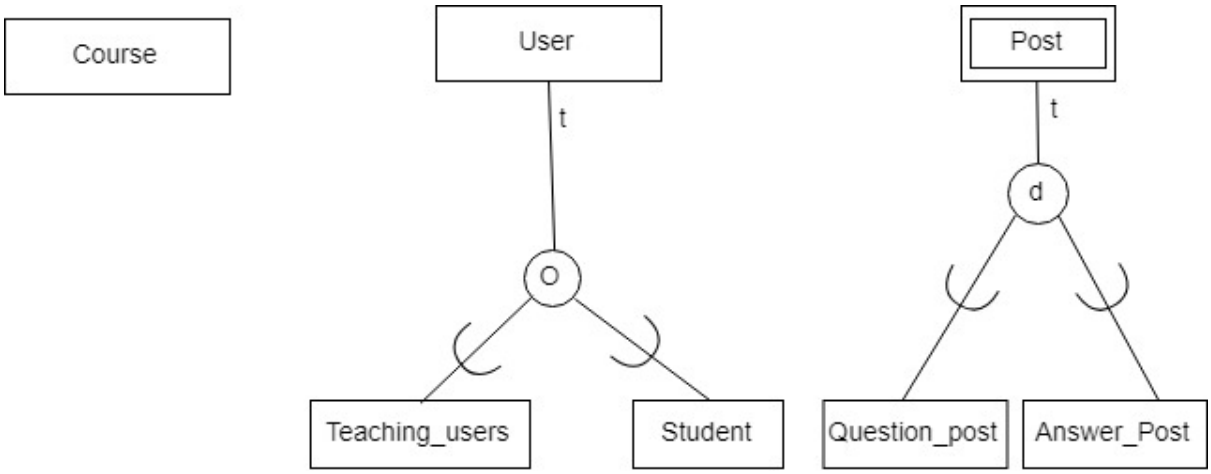
1. EER (Extended Entity Relation) Schema

1.1. Introduction

- The Extended Entity Relationship Diagram also called an EER model is a graphical representation of an information system that depicts relationships among people, objects, concepts etc. The EER diagram helps define business processes and is used as a foundation of a relational database.
- The following section of the project report provides you with an introduction to the ER schema for the project Tiny Piazza, which is a course forum website which provides functions such as User Management, Course Management, Post Management, User-Course Relationship Management and User-Post Relationship Management.

1.2. The Choice of Entities Types

- An Entity is something which represents a business concept with an apparent meaning to a particular set of users. The first step in the design of the EER diagram is to determine the entities. For designing the EER diagram for Tiny Piazza I have considered three main entities types: Course, User and Post.
- In the user and post entity types there are entities with distinguishing characteristics. So, I have defined the subclasses of User and Post Entity Types. For user entity type the subclasses are Teaching_user and Student Entity types. For post Entity Types the subclasses are Question_Post and Answer_Post Entity Types.

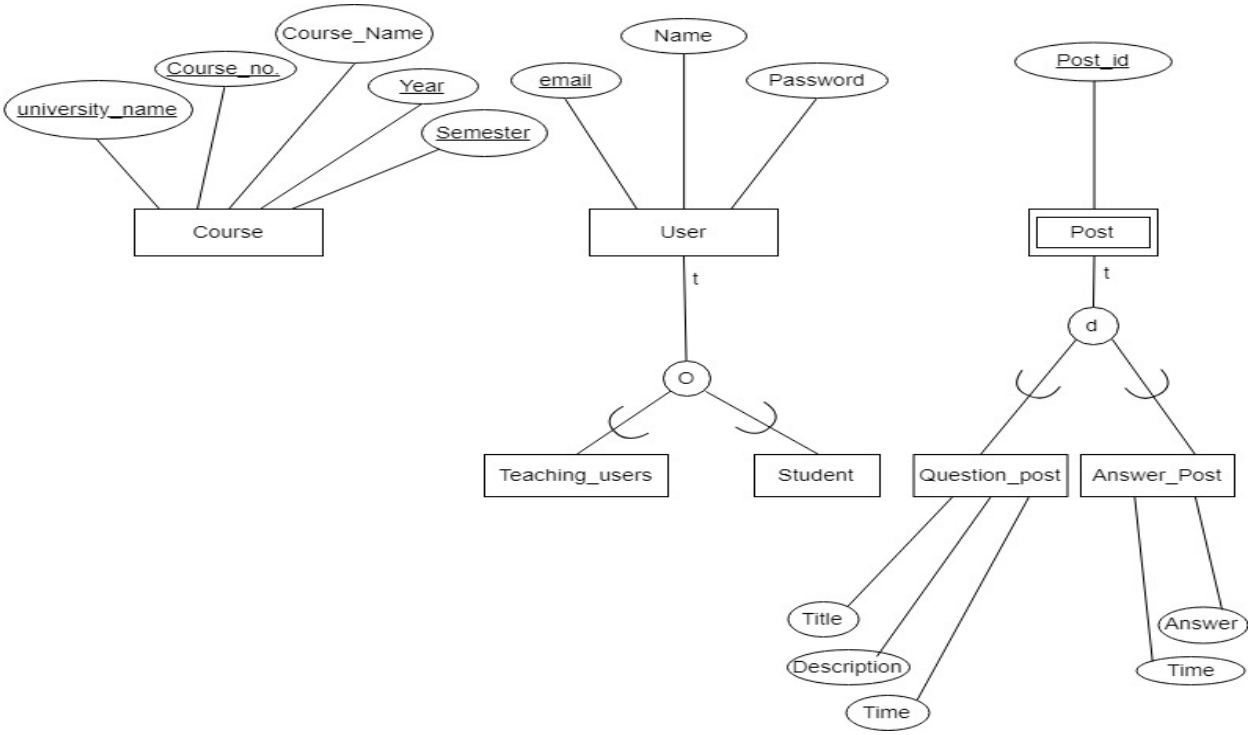


1.3. Attribute Types

- An attribute type represents a property of an entity type. So for the entities types represented above the attribute types are as follows:
- Attribute Type for Course are:

- University_Name
- Course_no
- Course_name
- Year
- Semester

- Attribute Type for User are:
  - Email
  - Name
  - Password
- Attribute Type for Post are:
  - Post_id
  - Creation_or_updation_id
- Attribute Type for Question_post are:
  - Title
  - Description
  - Time
- Attribute Type for Answer_Post are:
  - Answer
  - Time

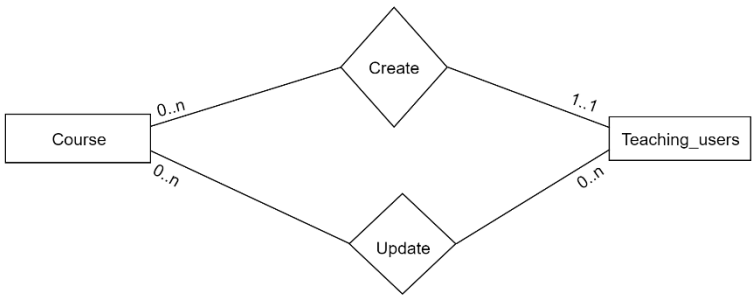


- In the figure above, the user superclass is specialized in subclasses teaching_user and Student and the post superclass is specialized into subclasses Question_post and Answer_post. Both the entity types, Teaching_user and Student inherit the attribute types email, name and password from entity type User and entity types Question_post and Answer_post inherit the attribute type post_id from entity type Post.

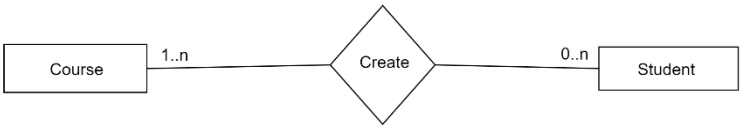
1.4. Relationship Types, Cardinalities and Degrees

All the relationships and cardinalities between entities for this project are explained below:

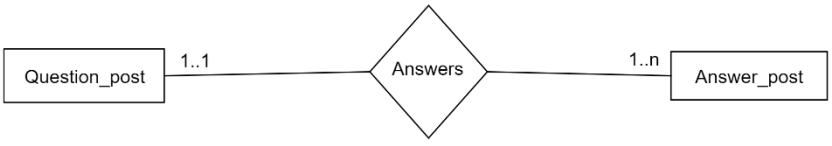
- There exists two relationships Create and Update between the entity types course and Teaching_user. A course could be created by only one teaching user but a teaching user could create zero to many courses. A course can be updated by zero to many teaching users and a teaching user can update zero to many courses.



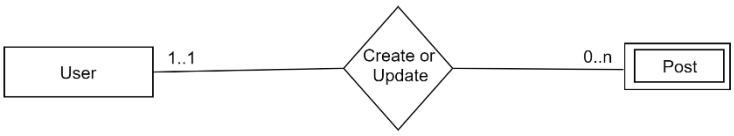
- There exists a relation type Register between entity types Course and Student. Each course can be registered by zero to many students and each student could be registered in one to many courses.



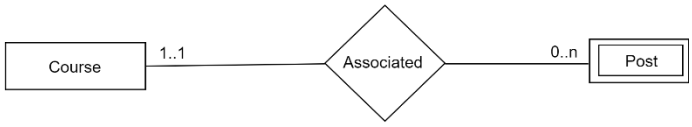
- There exists a relation type Answers between entity types Question_post and Answer_post. A question post could be answered by one to many Answer post and an answer post only answers one question post.



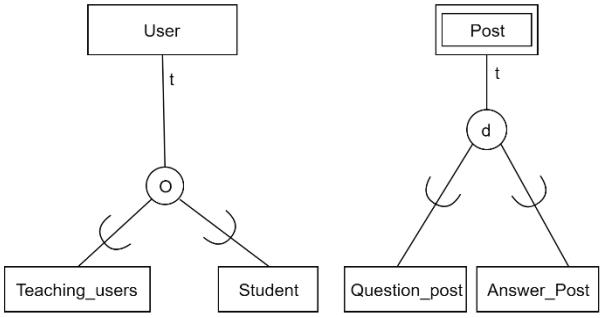
- There exists a relationship type Create or update between entity types User and Post. A user can create or update zero to many posts and a post can be created or updated by only one user.



- There exists a relationship type Associated Between entity types Course and Post. A course can be associated with zero or many posts but a post can only be associated with one course.



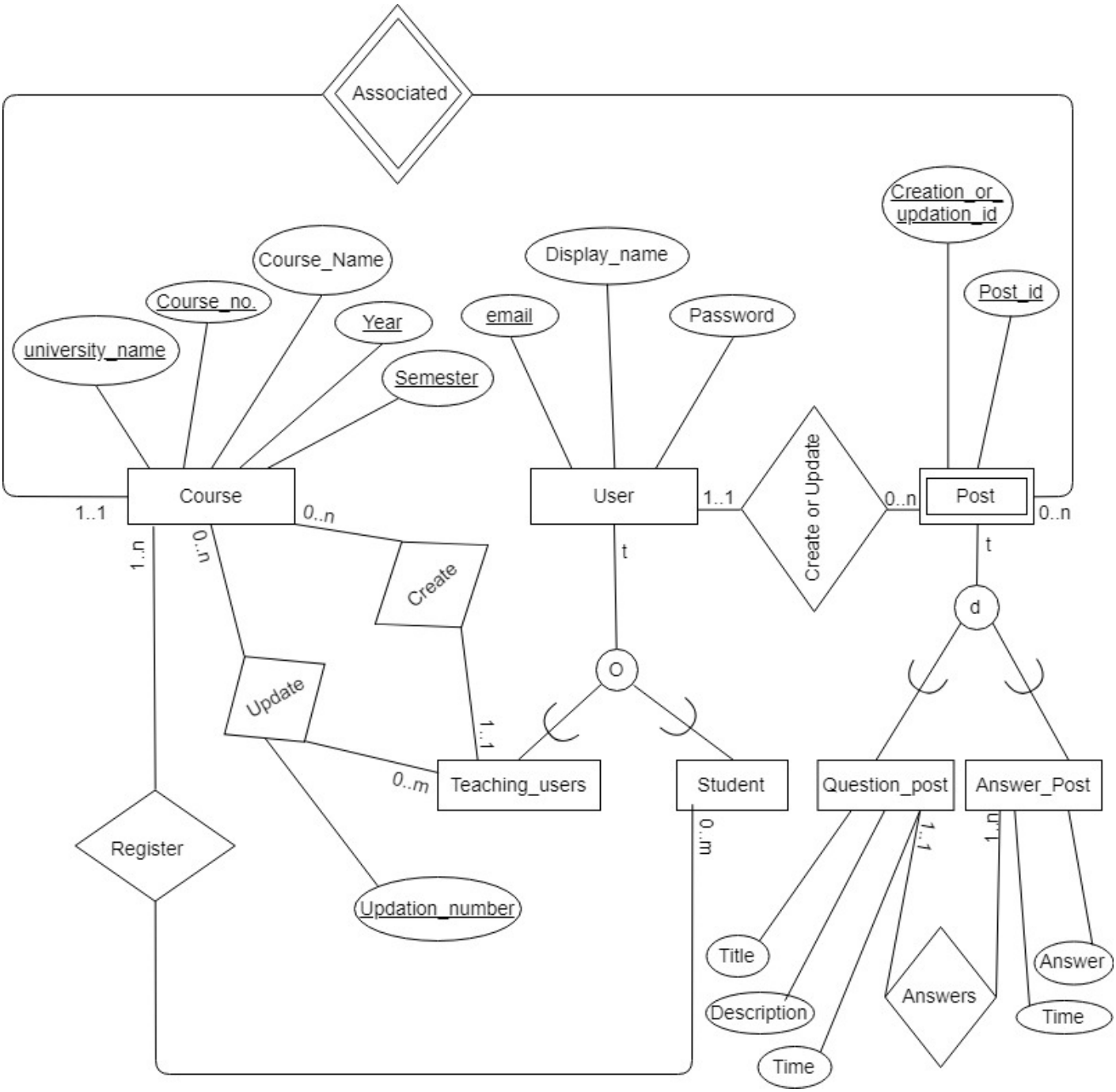
- There are two special types of relationships in the EER schema for this project called ‘IS-A relationship or Specialization. The User superclass is specialized in subclasses Teaching_user and Student. The small (o) notation specifies overlap specialization which means that a teaching user can be a student and a student can be a teaching user. The small (t) specifies total specialization which means that a user must be either teaching user or student or both. The Post superclass is specialized into subclasses Question_post and Answer_post. The small (d) notation specifies disjoint constraint which means that a question post cannot be an answer post and vice versa. The small (t) specifies total specialization which means that a post must be either question post or answer post, but it can not be both due to the disjoint constraint.



1.5. Identifying Weak Entity Types and Partial Keys

An entity type that does not have a key attribute type of its own is called a weak entity type. In our ER schema, the Post entity type with an attribute type post_id does not have its own set of key attribute types as post_id alone can not uniquely identify all the entities in the entity type Post. So, we make Post as weak entity type and Course as an owner entity type. Now, Post borrows all the attribute types from entity type Course. The attribute type Post_id of entity type post is called partial key.

1.6. The Extended Entity Relationship (EER) Diagram



1.7. Some questions on design choice with this ER schema

- **Why is the entity type Teaching_user is related to Course with Update and Create relationships and why not entity type course and user directly related?**
  This is because a course could be created and updated only by a teaching user and not by a student. The user entity type has user entities too. So, we can not connect course entity to the user entity.
- **Why the relationship between User and Post represented as a single relationship Create or Update but the relationships between Course and Teaching Users are connected using two different relationships Create and Update?**
  This is because the cardinalities between user and post entity types are similar for create and update, that is why it is represented as a single relationship, while the cardinalities for create and update relationships between the course and teaching user entity types are different. That is why it is represented as different relationships.
- **Why IS-A relationship is used between entity types Post, Question_post and Answer_post?**
  Question post and answer post are the type of post and is best represented by an IS-A relationship. Also, the disjoint and totality constraints can be best represented by an IS-A relationship. That is why we used an IS-A relationship.
- **Why did we add creation_or_updation_id and updation_number as attributes for post and update respectively?**
  If we do not keep a unique creation or updation number attribute type with the post entity type, it will create similar entities in the post table, as a single user can update a post multiple times. A similar thing would happen with the update table, if we do not keep the updation number because a teaching user can update a course multiple times.

2. Relational Database Schema

(Note: Before running the submitted .sql script in Mysql Workbench, please create a schema called tiny_piazza and then run the script.)

Steps for the conversion of EER schema into relational schema is given below:

- First, we convert the entity User into a relation called User with column names as email, display_name and password. Email is chosen as a primary key because it is unique and not null.
- Then the entity teaching_user is converted into relation teaching user with column names as email and display name. Email is chosen as the primary key because it is unique and not null. Email is the foreign keys and it references email from relation User.
- The entity student is converted into student relation with the column names email and display_names. Email is chosen as the primary key because it is unique and not null. Email is the foreign keys and it references email from relation User.
- The next relation is course with column names university_name, course_no, year, semester, course_name, creater_email. University_name, course_no, year, semester are chosen as the primary key. Creater_email is unique. Creater_email is a foreign key which references email from relation teaching_user.
- Since the relationship Update in the EER schema is a many to many relationship, it is converted to a relation called update in the relational schema. The column names are updation_number, university_name, course_no, year, semester, and updated_by_email. All the columns here is the set of primary key. Updated_by_email is a foreign key and it references email from the relation teaching_user. University_name, course_no, year and semester also foreign keys and it references and it references these columns from the relation course.
- Since the relationship Register in the EER schema is a many to many relationship, it is converted to a relation called register in the relational schema. The column names are university_name, course_no, year, semester and registered_by_email. All the column combined forms the primary key. University_name, course_no, year, semester are foreign keys and it references these column from relation course. Also, registered_email_address is a foreign key and it references email from relation student.
- The weak entity post from the EER schema is converted into a relation post in the relational schema. The table names are university_name, course_no, year, semester, post_id, posted_by_email and creation_or_updation_id. All the columns combined forms the primary key. University_name, course_no, year and semester are foreign

keys and are referenced from the relation course. Posted_by_email is also a foreign key and is referenced by email from relation user.

- The next relation is question_post relation. The column names are question_post_id, title, description and time. Question_post_id is chosen as the primary key. It is also a foreign key and it references post_id from the relation post.
- Another relation is answer_post. The column names are answer_post_id, answer and time. Answer_post_id is the primary key. It is also the foreign key and it references post_id from relation post.

I converted the EER schema into a relational schema by using the lecture notes and you can refer the .sql file for seeing the relational schema and further details.

3. Advantages and Disadvantages of this Design

3.1. Disadvantages

- In our schema, the user subclass such as Teaching Assistant and Professor for teaching user are represented. Since teaching assistant and professor data are not in separate relations it creates a barrier for updating of the database schema if any further requirement comes up from the client.
- Column comments is an industry-wide requirement and it is found missing in our design. Each table should have a column called comments. It should be populated if necessary, otherwise should be kept empty.
- Attribute types such as start_date and end_date for entity type course is not there. This creates a limitation for the usability of the database schema. Questions such as display all students name who were enrolled in spring 2018 in a particular course could not be answered with this schema.
- Industry requirement says that a primary key should be numeric but in our schema, it is sometimes not numeric. Example: email.
- Data is repeated in many tables and this creates update anomaly.
- There are certain database requirements which should be handled by the application. Example: the email_id should end with .edu.

3.2. Advantages

- The main advantage of our schema is that it is simple and if we want to remove the points stated in the disadvantages we will have to create many entity types which will create a very complex schema.

- As this is a simple database schema it results in low/no maintenance. It makes the changes easy and we need not take the system down for updates.
- Our schema correctly represents all the requirements in the requirement document.

4. Extra Credit Question

Consider the schema R(ABCD), the set of functional dependencies F = {A →BCD} and the decomposition (R1, R2) = (ABC, ABD).

Questions:

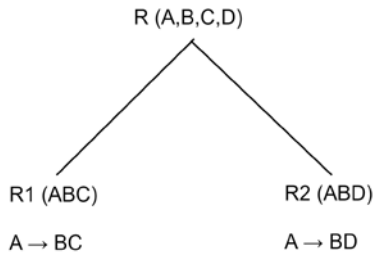
1. Are R1 and R2 in BCNF? Which dependencies are preserved by the decomposition?

Answer:

- Given, a relation R(ABCD), and the set of functional dependencies F = {A →BCD}
- First, we need to check that the given relation is in BCNF.
- For this, finding $A^+ = A$

  = ABCD (since, A →BCD)
- Since, A can functionally determine all the attributes of relationship R, therefore it is a candidate key. Now, according to the definition, a relationship R is said to be in Boyce-Codd normal form (BCNF) provided in each of its non-trivial functional dependency, the left part of the functional dependency is a candidate key or superset thereof.

  (Reference: This definition is directly referred from lecture slides.)
- So, the given relation R is in BCNF.
- Now, R is divided into R1 and R2 as shown in the figure below



- Now, let's check that R1 and R2 are in BCNF or not. In R1(ABC), A is still the candidate key so, it does not violate the BCNF rule. In R2(ABD), A is still the candidate key so, it also does not violates the BCNF rule. **So we can say that R1 and R2 are in BCNF.**
- Dependency Preserving Check:

- F.D. for a relational schema is said to be preserved after decomposition into a sub relational schemas R1 and R2, if we are able to obtain all the F.D. as were present initially from the sub relational schema.
- For a dependency preserving check whether we can derive F.D. of R from F.D. of R1 and R2.

  $A \rightarrow A$ ….(1)

  $A \rightarrow BC$ ….(2) (since, $A \rightarrow BC$ in R1)

  From (1) and (2), by using transitivity property we can say,

  $A \rightarrow BC$ …..(3)

  Also, from R2 we have $A \rightarrow BD$ …..(4)

  From (3) and (4), by using union property we have,

  $A \rightarrow BCD$

  Therefore, we can say that the decomposition of R into R1 and R2 is dependency preserving.

- Test for lossless join decomposition:

  A decomposition of relation R into R1 and R2 is lossless if

  $R1 \cap R2 \rightarrow R1$

  $R1 \cap R2 \rightarrow R2$

  In this question, $R1 \cap R2 = A$ which is a key of R1 and R2.

  Hence the given decomposition is lossless join. Lossless join is preserved by this decomposition.

2. If your answer to the previous question was "Both are in BCNF", how would you explain the following situation? An instance r of R contains single tuple (frank, 100K, mary, rochester). In the decomposition (R1, R2), the instance r1 of R1 consists of the tuple (frank, 100K, mary) and the instance r2 of R2 consists of the tuple (frank, 100K, rochester). The connection between frank and 100K is represented twice, creating a redundancy and a possible update anomaly. BCNF is supposed to prevent this. Justify your answers.

   Answer: I do not think that this condition is creating an updation anomaly. Sometimes data might repeat in a tables but it does not mean that the relational schema is not in BCNF. Since, A which is a candidate key is present in both the relational schemas R1

and R2 and we know that any candiate key could be taken as a primary key. So, we can say that all the touples in relations R1 and R2 would be unique.