Victoria University
of Bangladesh

Final Assessment

Md Bakhtiar Chowdhury

ID: 2121210061

Department: CSE

Semester: Summer -2022

Batch: 21th

Course Title: Database Management System

Course Code: CSI 221

Submitted To:

Md. Shahin Khan

Lecturer, Dept. of CSE/CSIT

Victoria University of Bangladesh

Submission Date: 08 October, 2022

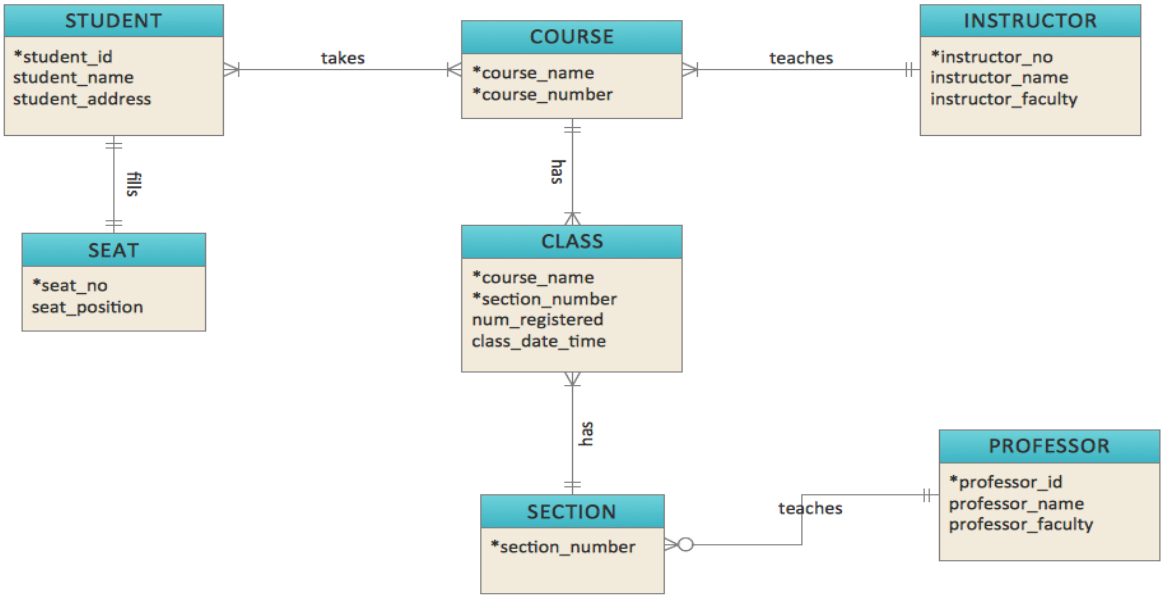
Answer to the question no 1(a)

 # Define ER Diagram . Give an example.

ER Diagram: An ER Diagram is a graphical representation of the relationships among people, objects, places, concepts or events within an information technology (IT) system.

- The ER data mode was developed to facilitate database design by allowing specification of an enterprise schema that represents the overall logical structure of a database.
- The ER model is very useful in mapping the meanings and interactions of real-world enterprises onto a conceptual schema
- The ER data model employs three basic concepts:
  entity sets,
  relationship sets,
  attributes.

For Example: A student (student_id, student_name, student_Address) takes Course (Course_name, Course_number) of a Instructor (Instructor_no, Instructor_name, Instructo_faculty). Here, we have an entity relationship between Student, Course, Teacher.



Answer to the question no 1(b)

 # What are the database proceeds (abstract) in two final design phases? Explain with example.

There are three main phases of a retaliation database design. The two database proceeds (abstract) before final design are:

- Conceptual Design.
- Logical Design.

Conceptual Design: Conceptual database design is part of the database design process, which consists of the activities of gathering requirements, analyzing the requirements and incorporates security constraints. The purpose of creating a conceptual data model is to establish entities, their attributes, and relationships.

For example: In a Sales Database, Customer and Product are two entities and sales are their relationship. Here, Customer number and name are attributes of the Customer entity and Product name and price are attributes of product entity. Sale is the relationship between the customer and product.

Logical Design: The Logical Data Model is used to define the structure of data elements and to set relationships between them. The logical data model adds further information to the conceptual data model elements. The advantage of using a Logical data model is to provide a foundation to form the base for the Physical model. It defines HOW the system should be implemented regardless of the DBMS. This model is typically created by Data Architects and Business Analysts. The purpose is to developed technical map of rules and data structures. At this level, no primary or secondary key is defined.

For Example: In our previous Sales Database, there is a many-to-many relationship between Products and Customers.

Answer to the question no 1(c)

 # Write down the list of four Entity-Relationship Model.

Answer

The entity-relationship (E-R) data model was developed to facilitate database design by allowing the specification of an enterprise schema that represents the overall logical structure of a database. The E-R model is very useful in mapping the meanings and interactions of real-world enterprises onto a conceptual schema. Because of this usefulness, many database-design tools draw on concepts from the E-R model. The E-R data model employs three basic concepts: entity sets, relationship sets, and attributes, which we study first. The E-R model also has an associated diagrammatic representation, the E-R diagram, which we study later in this chapter

- **Entity Sets**
  An entity is a “thing” or “object” in the real world that is distinguishable from all other objects. For example, each person in a university is an entity. An entity has a set of properties, and the values for some set of properties may uniquely identify an entity. For instance, a person may have a person id property whose value uniquely identifies that person. Thus, the value 677-89-9011 for person id would uniquely identify one particular person in the university. Similarly, courses can be thought of as entities, and course id uniquely identifies a course entity in the university. An entity may be concrete, such as a person or a book, or it may be abstract, such as a course, a course offering, or a flight reservation

76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

- **Relationship Sets**

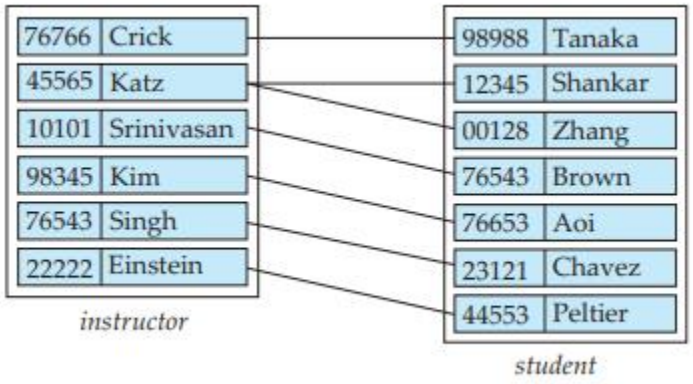
A relationship is an association among several entities. For example, we can define a relationship advisor that associates instructor Katz with student Shankar.

This relationship specifies that Katz is an advisor to student Shankar. A relationship set is a set of relationships of the same type. Formally, it is a mathematical relation on n ≥ 2 (possibly nondistinct) entity sets. If E1, E2,..., En are entity sets, then a relationship set R is a subset of

$$\{(e1,e2,...,en) \mid e1 \in E1, e2 \in E2, ..., en \in En\}$$

where (e1,e2,...,en) is a relationship.

Consider the two entity sets instructor and student



The list of four Entity-Relationship Model:

- One-to-One Relationships
- One-to-Many Relationships
- May to One Relationships
- Many-to-Many Relationships

One-to-One Relationships: One entity from entity set X can be associated with at most one entity of entity set Y and vice versa. Example: One student can register

for numerous courses. However, all those courses have a single line back to that one student.

One-to-many: One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity. For example, one class is consisting of multiple students.

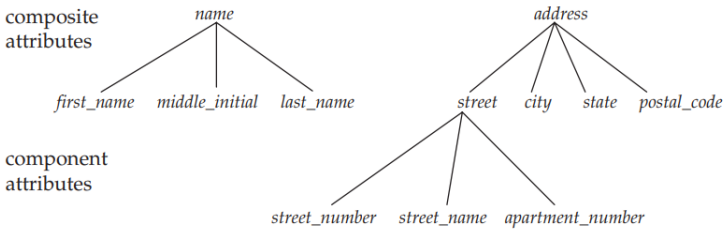
Many to One: More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X. For example, many students belong to the same class.

Many to Many: One entity from X can be associated with more than one entity from Y and vice versa. For example, Students as a group are associated with multiple faculty members, and faculty members can be associated with multiple students.

- **Attributes**

For each attribute, there is a set of permitted values, called the domain, or value set, of that attribute. The domain of attribute course id might be the set of all text strings of a certain length. Similarly, the domain of attribute semester might be strings from the set {Fall, Winter, Spring, Summer}. Formally, an attribute of an entity set is a function that maps from the entity set into a domain. Since an entity set may have several attributes, each entity can be described by a set of (attribute, data value) pairs, one pair for each attribute of the entity set. For example, a particular instructor entity may be described by the set {(ID, 76766), (name, Crick), (dept name, Biology), (salary, 72000)}, meaning that the entity describes a person named Crick whose instructor ID is 76766, who is a member of the Biology

department with salary of $72,000. We can see, at this point, an integration of the abstract schema with the actual enterprise being modeled. The attribute values describing an entity constitute a significant portion of the data stored in the database.



Answer to the question no 3(a)

 # What is Normalization ? Write down the Goals of Normalization.

Normalization: Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency. Data normalization is an essential process for professionals that deal with large amounts of data. If the database is not organized and normalized, something as small as one deletion in a data cell can set off a sequence of errors for other cells throughout the database.

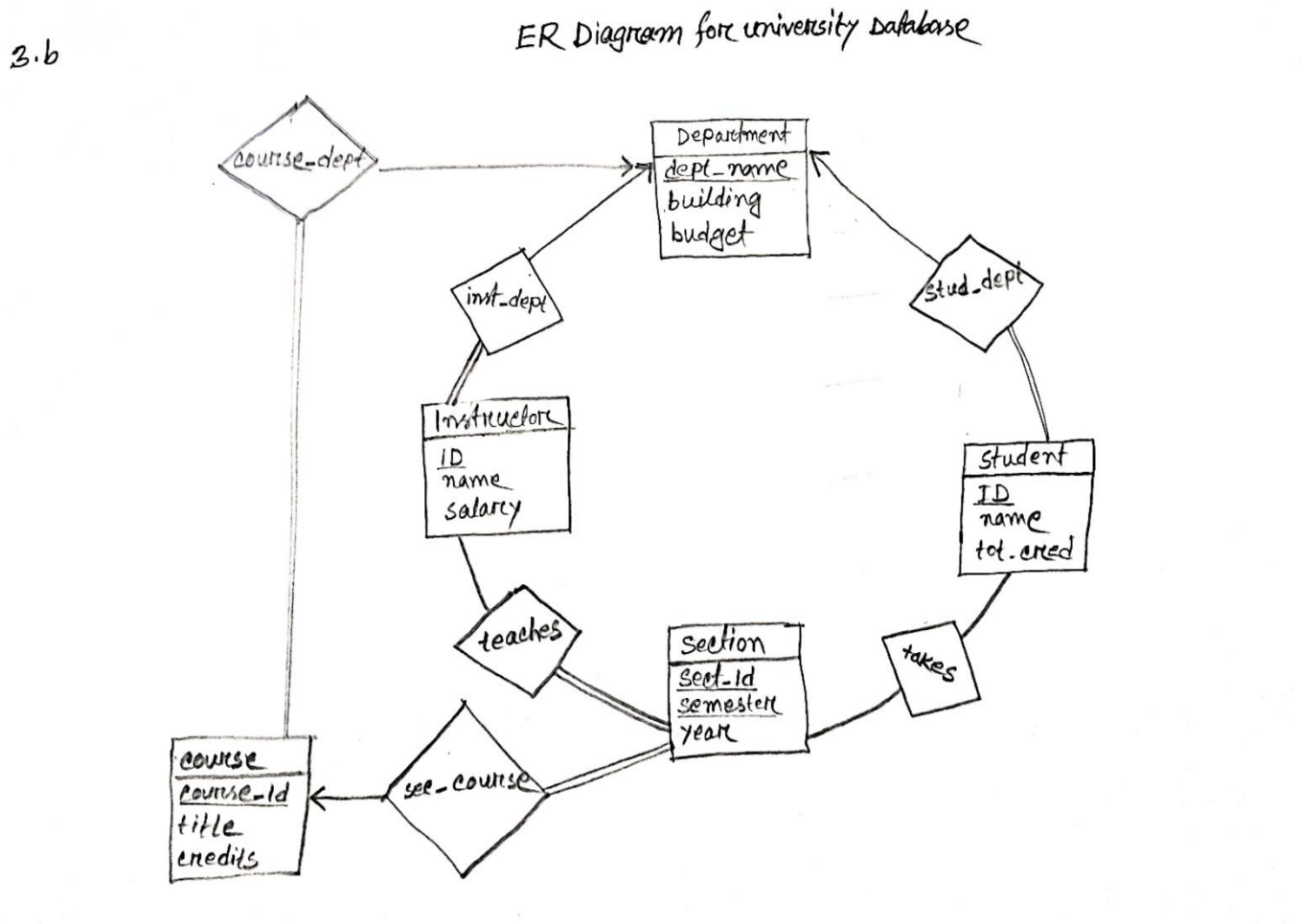
The main objective of database normalization is to eliminate redundancy, minimize data modification errors, and simplify the query process.

The Goals of Normalizations are:

- **Improved overall database organization:** With normalization, database will be more structured and well defined. It will help us minimizing the organizational entities, duplication and location errors as well as the database can be updated easily.
- **Data consistency:** Consistent data is crucial for all teams within a business to stay on the same page. Data normalization will ensure consistency across development, research, and sales teams. Consistent data will also improve workflow between departments and align their information sets.
- **Reduces redundancy:** Redundancy is a commonly overlooked data storage issue. Reducing redundancy will ultimately help reduce file size and therefore speed up analysis and data processing time.
- **Cost reduction:** Cost reduction due to normalization involves reduced redundant data. For example, if file size is reduced, data processors will be faster and will consume less storage. Moreover, workflow will be smoother due to consistency of database. Organization can ensure concurrent access which will save time for other necessary tasks. So, the overall performance of the organization will be improved.
- **Increased security:** Because normalization requires that data is more accurately located and uniformly organized, security is significantly increased.

Answer to the question no 3(b)

 # Construct ER Model Diagram for "University Database" (entity: takes, student, department, advisor, course, instructor)



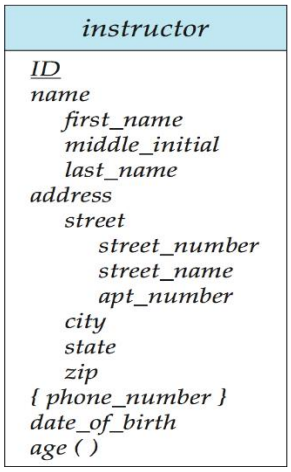
Answer to the question no 4(a)

 # Define with example of Composite Attributes.
Answer:

composite attributes : A composite attributes is a combination of other attributes. For example, Address attribute consists of Street, City, State, and Country. In ER diagram, composite attribute is represented by an oval comprising of ovals.

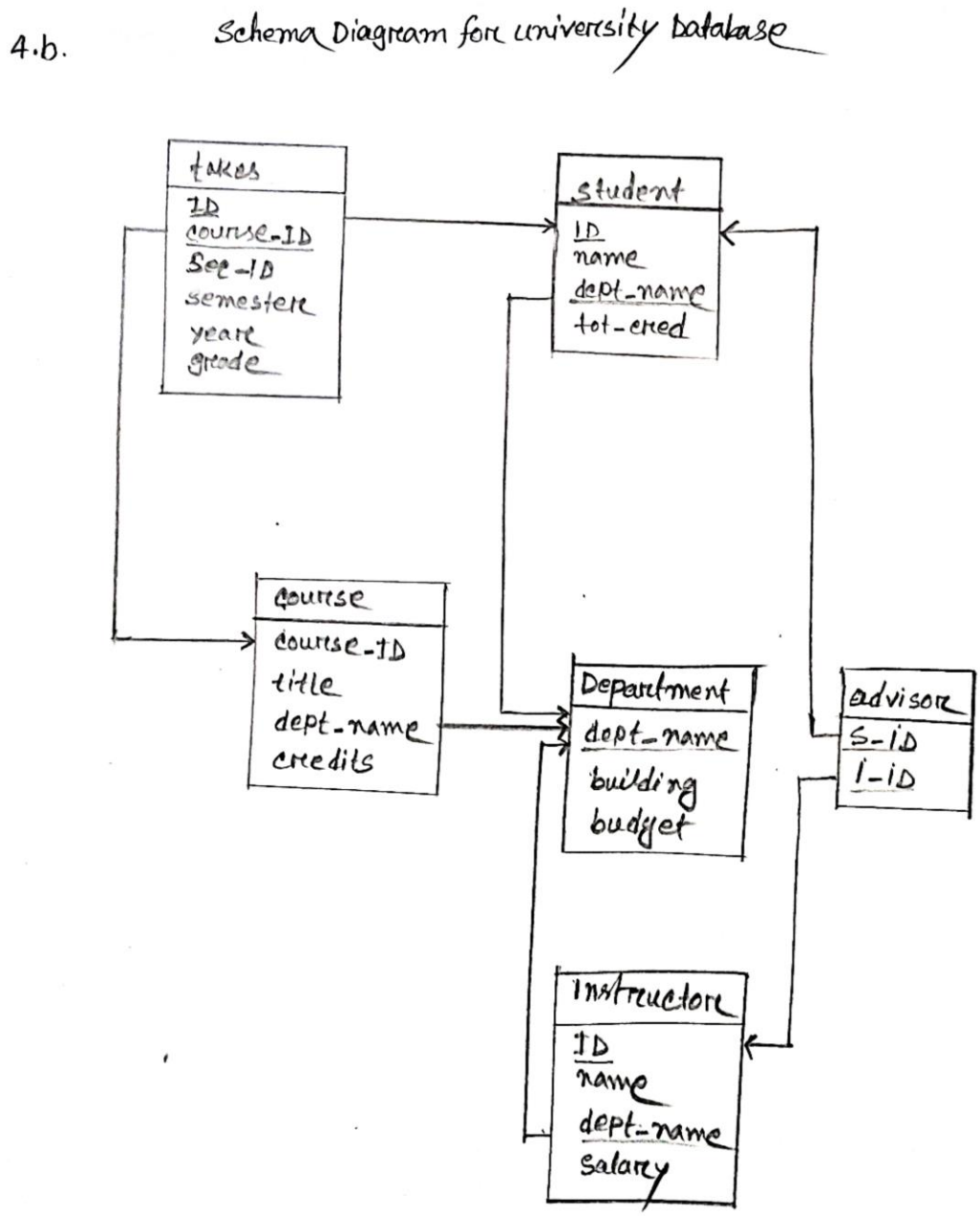
- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - Example: given entity set instructor with composite attribute name with component attributes first_name and last_name the schema corresponding to the entity set has two attributes name_first_name and name_last_name
    - Prefix omitted if there is no ambiguity (name_first_name could be first_name)
- Ignoring multivalued attributes, extended instructor schema is

instructor(ID,
first_name, middle_initial, last_name,
street_number, street_name,
apt_number, city, state, zip_code,
date_of_birth)



Answer to the question no 4(b)

 # Construct Schema Diagram for "University Database" (entity: takes, student, department, advisor, course, instructor)



Answer to the question no 5(a)

 # Suppose VUB has a university database where the entity name is Instructor (Lid, Name, Dept_name, Salary) ,Student (Sid,name,sec,) and Course (Cid , sem,year) .

 # Find the names of all instructors whose name includes the substring "han".

```
select Name
from instructor
where name like '%han%'
```

 # Find out the name who has beginning with '_Intro'

```
select name
from instructor
where name like '_Intro%'
```

 # Find courses that ran in Fall 2022 but not in Spring 2022

```
(select C_id from Course where sem = 'Fall' and year = 2022)
 except
(select C_id from Course where sem = 'Spring' and year = 2022)
```

Find the salaries of all instructors that are less than the largest salary.

```
select distinct T.salary
from instructor as T, instructor as S
where T.salary < S.Salary
```

 # Find the list in alphabetic order the names of all instructors.

```
select distinct name
from  instructor
order by name asc
```

>>>>END<<<<