Autumn 2022

1(a)

Instance:

A collection of information that stored in a database at a particular moment is called an instance of the database.

Schema:

The overall design of the database is called database schema. Schema will not be changed frequently. It is the logical structure of a database. It does not show the data in the database.

Storage Manager:

Storage Manager is a program that provides an interface between the data stored in the database and the queries received. It is also known as Database Control System. It maintains the consistency and integrity of the database by applying the constraints and executing the DCL statements. It is responsible for updating, storing, deleting, and retrieving data in the database.

It contains the following components –

- **Authorization Manager:** It ensures role-based access control, i.e,. checks whether the particular person is privileged to perform the requested operation or not.
- **Integrity Manager:** It checks the integrity constraints when the database is modified.
- **Transaction Manager:** It controls concurrent access by performing the operations in a scheduled way that it receives the transaction. Thus, it ensures that the database remains in the consistent state before and after the execution of a transaction.
- **File Manager:** It manages the file space and the data structure used to represent information in the database.
- **Buffer Manager:** It is responsible for cache memory and the transfer of data between the secondary storage and main memory.

1(b)

In database systems, **atomicity** is one of the ACID (Atomicity, Consistency, Isolation, Durability) transaction properties. An **atomic transaction** is an indivisible and irreducible series of database operations such that either all occurs, or nothing occurs.

An example of an atomic transaction is a monetary transfer from bank account A to account B. It consists of two operations, withdrawing the money from account A and saving it to account B. Performing these operations in an atomic transaction ensures that the database remains in a consistent state, that is, money is neither lost nor created if either of those two operations fails.

Prepared by – Sadman Hasan & Abdullah Al Azad

1(b)(or)

Primary key: The primary key of a relational table uniquely identifies each record in the table.

Example:

The ideal primary key for a table of students would be their **ID number**, as this would uniquely identify each student in the table.

Super key: A super key is a set of one or more keys that are used to identify data or records uniquely in a database table.

Example:

Email: If each customer's email address is unique, the Email attribute can be a superkey.
First_Name and Last_Name: If no two customers have the same first and last name, the combination of First_Name and Last_Name attributes can be a superkey.

Candidate key: A super key such that no proper subset is a super key within relation.

Example:

Emp_Id: An attribute that stores the value of the employee identification number.
Emp_name: An attribute that stores the name of the employee holding the specified employee id.
Emp_email: An attribute that stores the email id of the specified employees.

Foreign key: An attribute or set of attributes within one relation that matches the candidate key of a relation.

Example:

In a table of orders, a foreign key could be a reference to the primery key of the customers table, indicating which customer placed the order.

1(c)

A Data Model in Database Management System (DBMS), is the concept of tools that are developed to summarize the description of the database.

OR,

Data model is collection of conceptual tools for describing data, data relationships, data semantics and consistency constraints. It provides a way to describe the design of a database at all levels.

OR,

Data models are often used as an aid to communication between the business people defining the requirements for a computer system and the technical people defining the design in response to those requirements.

Prepared by – Sadman Hasan & Abdullah Al Azad

Types of model -

Entity-Relationship Model

Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them.

Relational Model

The most popular data model in DBMS is the Relational Model. It is more scientific a model than others.

Object-base Data model

Object-base Data model is a model which extends the E-R model with notaions of encapsulation and object identity.

Semi-structured date model

Semi-structured date model permits the specification of data where individual data items of the same type may have different sets of attributes.

DDL VS DML:

DDL	DML
It stands for Data Definition Language.	It stands for Data Manipulation Language.
It is used to create database schema and can be used to define some constraints as well.	It is used to add, retrieve or update the data.
It basically defines the column (Attributes) of the table.	It add or update the row of the table. These rows are called as tuple.
It doesn't have any further classification.	It is further classified into Procedural and Non-Procedural DML.
Basic command present in DDL are CREATE, DROP, RENAME, ALTER etc.	BASIC command present in DML are UPDATE, INSERT, MERGE etc.
DDL does not use WHERE clause in its statement.	While DML uses WHERE clause in its statement.

2(a)

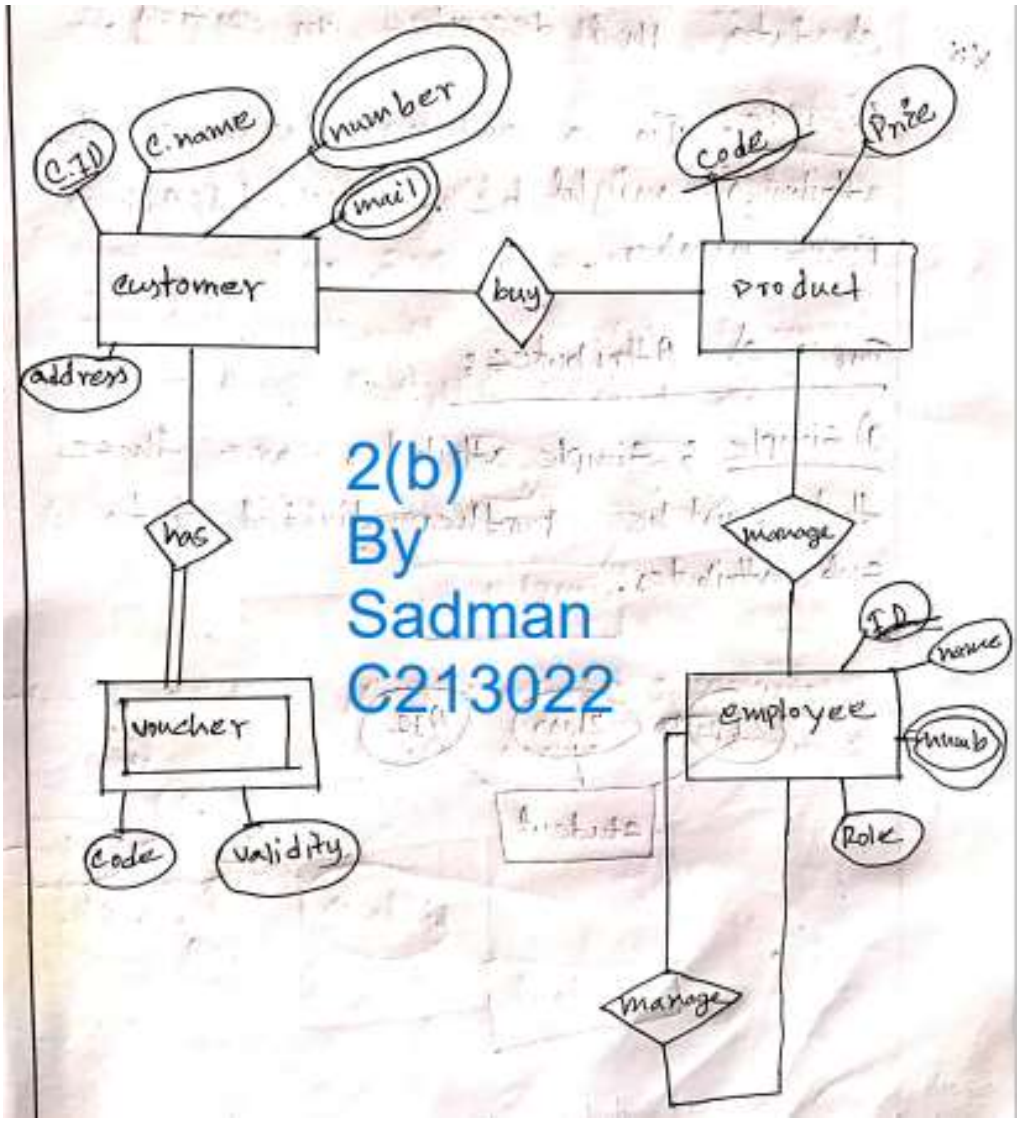
Cardinality: cardinality represents the number of times an entity of an entity set participates in a relationship set.

- one-to-one
- one-to-many
- many-to-one
- many-to-many

Total Participation – Each entity is involved in the relationship. Total participation is represented by double lines.

Partial participation – Not all entities are involved in the relationship. Partial participation is represented by single lines.

2(b)



Prepared by – Sadman Hasan & Abdullah Al Azad

2(c)

Attribute: In a database management system (DBMS), an attribute is a piece of data or properties that describes an entity.

Types of Attributes

1. Simple Attributes

Simple attributes are those that cannot be further divided into sub-attributes.
Example:
A student's roll number of a student or the employee identification number.

2. Composite Attributes

Composite attributes are made up of two or more simple attributes.

Example:
a person's address may be a composite attribute that is made up of the person's street address, city, state, and zip code.

3. Single Valued Attributes

Single-valued attributes can only have one value.

Example:
a person's Social Security Number is a single-valued attribute.

4. Multivalued Attributes

Multivalued attributes can have more than one value.

Example:
a person may have multiple email addresses or phone numbers.

5. Derived Attributes

Derived attributes are based on other attributes and are not stored directly in the database.

Example:

Prepared by – Sadman Hasan & Abdullah Al Azad

Consider a database of employees. Each employee has a date of birth, and we might want to calculate their age. However, age is a derived attribute because it can be determined from the date of birth. As such, it would not make sense to store it directly in the database.

6. Complex Attributes

The complex attribute in DBMS involves both multivalued and composite attributes.

Example:
someone might have more than one house, and each house might have more than one phone. The phone is then considered a complex attribute.

7. Stored Attributes

In a DBMS, stored attributes are the data that remain constant and fixed for an entity instance. These values help in deriving the derived attributes.

Example:

8. **Null Attributes**

Example:

3(a)

(i) $\prod_{customer\_name}(depositor) \cup \prod_{customer\_name}(borrower)$

(ii) $\prod_{customer\_name}(\sigma_{branch\_name="Kumira" \text{ and } borrower.loan\_number = loan.loan\_number}(borrower \times loan))$

(iii) $\prod_{customer\_name}(\sigma_{branch\_name="GEC"}(loan)) - \prod_{customer\_name}(\sigma_{branch\_name="GEC"}(account))$

(iv) $\prod_{account\_name}(\sigma_{balance>=1000000}(account))$

Prepared by – Sadman Hasan & Abdullah Al Azad

3(b)

Relational Algebra is a procedural query language, it is used to provide a single table / relation as output of performing operations on more than one relations.

- **Selection operation (σ)**
  The selection operator denoted by sigma σ is used to select the tuples of a relation based on some condition. Only those tuples that fall under certain conditions are selected.
  Syntax
  σ branch-name="GEC"(loan)

- **Projection operation (∏)**
  The projection operator denoted by ∏ is used to select columns from a specific reaction. Only specific columns are selected.
  Syntax

  ∏column1 , column2 , … , column (relation_name)

- **Cross Product(X)**
  Cross product is denoted using the X symbol and is used to find the value of join of two variables. In cross product each tuple of relation1 is crossed with each tuple of relation2. Which makes the output relation of the order nXm, where n is the number of tuples in relation1 and m is the number of tuples in relation2.
  Syntax

  relation1 X relation2

- **Union (U)**
  The union of two relations relation1 and relation2 will gives the tuples that are either in relation1 or in relation2 but tuples that are in both relation1 and relation2 are considered only once.
  Also both relations should be of the same domain for finding there union.
  Syntax

  relation1 U relation2

- **Set difference (-) operator**
  Set difference operator is denoted by - symbol. Relation1 - relation2 will result into a relation in which the tuple in relation1 and not in relation2 are present. For calculating minus too, the relations must be union compatible.
  Syntax

  relation1 - relation2

- **rename(ρ)**

Prepared by – Sadman Hasan & Abdullah Al Azad

The rename operation denoted by the ρ is used to rename the given relation to another name given.

Syntax

ρ(new_name , old_name)

- **Join operation**

  It combines R1 and R2 with respect to a condition. It is denoted by ⋈.

  Joins are of different types

  1. Inner Join -
     - Theta join.
     - Natural join.
     - Equi Join
  2. Outer join −
     - Left outer join.
     - Right outer join.
     - Full outer join.

- **Division operation**

  The division operator is used for queries which involves the 'all'

  Syntax

  R1/R2 = tuples of R1 associated with all tuples of R2.

- **Intersection**

  An intersection is defined by the symbol ∩. Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.

  Syntax

  A ∩ B

3(b)(or)

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

Mapping ER diagram to database-

1. Create a separate relational table for each entity.

   This a logical starting point when mapping an E-R diagram into a relational database model. It is generally useful first to specify the database schema before proceeding to expansion of the relations to account for specific tuples.

2. Determine the primary key for each of the relations.

The primary key must uniquely identify any row within the table.

3. Determine the attributes for each of the entities.
   A complete E-R diagram includes specification of all attributes, including the key attribute.
4. Implement the relationships among the entities.
   This is accomplished by ensuring that the primary key in one table also exists as an attribute in every table (entity)for which there is a relationship specified in the entity-relationship diagram.

   - One-to-many (1:N or N:1)
   - One-to-one (1:1)
   - Many-to-many (M:N)

5. Determine the attributes, if any, for each of the relationship tables.

Prepared by –
Sadman Hasan
Abdullah Al Azad