Academic year 2024-2025 (Odd Sem)

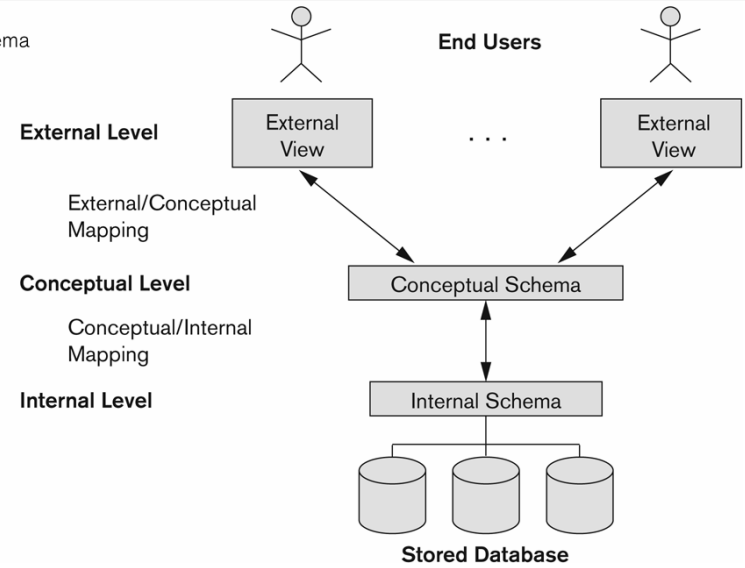
DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

SCHEME AND SOLUTION

Date	24th November 2024	**Maximum Marks**	10+50
Course Code	CD252IA	**Duration**	90 Minutes
Sem	V	Faculty:	

Database Management Systems (Common to CS, IS, CD, AI & CY)

Sl.No	PART-A	M	BT	CO
1.	**Tasks of Actors on the Scene:** 1. Querying the database for specific information (e.g., retrieving customer data). 2. Updating or modifying records in the database (e.g., editing employee information). Examples: Data Analysts, Bank Clerks, Sales Executives. **Tasks of Workers Behind the Scene:** 1. Ensuring the database system is running efficiently (e.g., tuning database performance). 2. Implementing security measures to protect the database (e.g., setting user access permissions). Examples: Database Administrators (DBAs), System Engineers, Security Specialists.	**2**	**L3**	**1**
2.	**Representational (or Implementation) Data Model**. Example: Model which organizes data into tables (relations) with rows and columns, allowing users to query and manipulate data using SQL.	**2**	**L3**	**1**
3.	It depends on the combination of the **Member ID** (from the **Member** entity) and the **Book ID** (from the **Book** entity) for its identification.	**2**	**L3**	**2**
4.	Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. Multivalued attributes are those for which different users may have different values for the same attribute. It may have lower and upper bounds to constrain the number of values allowed for each individual entity. – 1 mark Representation for both: 1 mark	**2**	**L2**	**2**
5.	In some cases, a particular entity may not have an applicable value for an attribute. For such situations, a special value called NULL is created. An address of a single-family home would have NULL for its Apartment_number attribute, and a person with no college degree would have NULL for College_degrees. NULL can also be used if we do not know the value of an attribute for a particular entity.	**2**	**L2**	**1**
Sl. No.	**PART-B**	**M**	**BT**	**CO**
1	A relationship type R among n entity types E1, E2, ..., En defines a set of associations—or a relationship set—among entities from these entity types. Mathematically, the relationship set R is a set of relationship instances ri, where each ri associates n individual entities (e1, e2, ..., en), and each entity ej in ri is a member of entity set Ej, $1 \leq j \leq n$. Hence, a relationship set is a mathematical relation on E1, E2, ..., En; alternatively, it can be defined as a subset of the Cartesian product of the entity sets E1 × E2 × ... × En. Each of the entity types E1, E 2, ..., En is said to participate in the relationship type R; similarly, each of the individual entities e1, e2, ..., en is said to participate in the relationship instance ri = (e1, e2, ..., en). – 040 marks iii) An example of a ternary relationship is SUPPLY, where each relationship instance ri associates three entities—a supplier s, a part p, and a project j—whenever s supplies part p to project j. Relationships can generally be of any degree, but the ones most common are binary relationships. iv) Relationships as Attributes: Consider the WORKS_FOR relationship type. One can think of an attribute called Department of the EMPLOYEE entity type, where the value of Department for each EMPLOYEE entity is (a reference to) the DEPARTMENT entity for which that employee works. Hence, the value set for this	**10**	**L2**	**1**

		Department attribute is the set of all DEPARTMENT entities, which is the DEPARTMENT entity set. v) Role names are not technically necessary in relationship types where all the participating entity types are distinct, since each participating entity type name can be used as the role name. However, in some cases the same entity type participates more than once in a relationship type in different roles. In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays. Such relationship types are called recursive relationships. – 06 marks			
2	**(a)**	**Figure 2.2** The three-schema architecture.  Fig -2 marks, Expln: 4 marks	**06** **04**	**L2** **L2**	**1** **1**
	(b)	**Logical data independence** is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item). **Physical data independence** is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized—for example, by creating additional access structures—to improve the performance of retrieval or update. 2X2 =4			
3	**(a)**	Self describing nature Insulation between program and data, and data abstraction Support of multiple views of the data Sharing of data and multiuser transaction processing (any 3 points)	**06** **04**	**L3** **L2**	**1** **1**
	(b)	It may be more desirable to use regular files under the following circumstances: ▪ Simple, well-defined database applications that are not expected to change at all ▪ Stringent, real-time requirements for some application programs that may not be met because of DBMS overhead ▪ Embedded systems with limited storage capacity, where a general-purpose DBMS would not fit ▪ No multiple-user access to data			
4	**(a)**	**Entities and Attributes** 1. **Farmer** ○ **Attributes**: ▪ FarmerID (Primary Key) ▪ Name	**10**	**L4**	**2**

		▪ ContactNumber ▪ Address ▪ BankDetails 2. **Consumer/Wholesaler** ○ **Attributes**: ▪ ConsumerID (Primary Key) ▪ Name ▪ ContactNumber ▪ Address 3. **Crop/Produce** ○ **Attributes**: ▪ CropID (Primary Key) ▪ CropName ▪ QuantityAvailable ▪ PricePerUnit 4. **Order** ○ **Attributes**: ▪ OrderID (Primary Key) ▪ OrderDate ▪ QuantityOrdered ▪ TotalPrice ▪ PaymentStatus 5. **Transaction** ○ **Attributes**: ▪ TransactionID (Primary Key) ▪ TransactionDate ▪ FarmerID (Foreign Key) ▪ ConsumerID (Foreign Key) ▪ OrderID (Foreign Key) **Relationships** 1. **Farmer uploads Crop/Produce** ○ **Relationship**: One Farmer can upload multiple Crops, but each Crop belongs to one Farmer. ○ **Type**: One-to-Many 2. **Consumer places Order** ○ **Relationship**: One Consumer can place multiple Orders, but each Order is placed by one Consumer. ○ **Type**: One-to-Many 3. **Crop is part of Order** ○ **Relationship**: One Order can contain multiple Crops, and each Crop can be part of multiple Orders. ○ **Type**: Many-to-Many (resolved through an associative entity, e.g., **OrderDetails**) 4. **Farmer and Consumer/Wholesaler involved in Transaction** ○ **Relationship**: Each Transaction involves one Farmer and one Consumer. ○ **Type**: Many-to-Many (can be resolved using the **Transaction** entity). **ER Diagram – 07 marks** **Cardinality ratio – 03 marks**			
5	**(a)**	**Relational Tables – 07 marks** 1. **Farmer** ○ FarmerID (Primary Key)	10	4	L2

- Name
- ContactNumber
- Address
- BankDetails

2. **Consumer**
   - ConsumerID (Primary Key)
   - Name
   - ContactNumber
   - Address
3. **Crop**
   - CropID (Primary Key)
   - FarmerID (Foreign Key, references Farmer.FarmerID)
   - CropName
   - QuantityAvailable
   - PricePerUnit
4. **Order**
   - OrderID (Primary Key)
   - ConsumerID (Foreign Key, references Consumer.ConsumerID)
   - OrderDate
   - PaymentStatus
5. **OrderDetails** (Associative table for Many-to-Many relationship between Order and Crop)
   - OrderID (Foreign Key, references Order.OrderID)
   - CropID (Foreign Key, references Crop.CropID)
   - QuantityOrdered
   - TotalPrice
6. **Transaction**
   - TransactionID (Primary Key)
   - FarmerID (Foreign Key, references Farmer.FarmerID)
   - ConsumerID (Foreign Key, references Consumer.ConsumerID)
   - OrderID (Foreign Key, references Order.OrderID)
   - TransactionDate

Relational Schema Diagram Representation – 03 marks

- **Primary Keys (PK)**: Underlined in each table.
- **Foreign Keys (FK)**: Marked as references to their respective tables.
- Relationships are shown using lines/arrows connecting FK columns to their referenced PK columns.

Representation of the above diagram step by step as per algorithm