Modified

CBCS SCHEME

USN | | | | | | | | | | |

BCS403

Fourth Semester B.E./B.Tech. Degree Examination, June/July 2024
Database Management Systems

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

		Module – 1	M	L	C
Q.1	a.	Define database. Elaborate component modules of DBMS and their interactions.	10	L2	CO1
	b.	Describe the three-schema architecture. Why do we need mappings among schema levels?	06	L2	CO1
	c.	Explain the difference between logical and physical data independence.	04	L2	CO1
		OR			
Q.2	a.	Draw an ER diagram for an COMPANY database with employee, department, project as strong entities and dependent as weak entity. Specify the constraints, relationships and ratios in the ER diagram.	10	L3	CO3
	b.	Define the following terms with example for each using ER notations: Entity, attribute, composite attribute, multivalued attribute, participation role.	10	L3	CO3
		Module – 2			
Q.3	a.	Discuss the update operations and dealing with constraint violations with suitable examples.	08	L2	CO2
	b.	Illustrate the relational algebra operators with examples for select and project operation.	06	L2	CO2
	c.	Discuss the characteristics of relations that make them different from ordinary table and files.	06	L2	CO2
		OR			
Q.4	a.	Perform (i) Student U instructor (ii) Student ∩ Instructor (iii) Student – Instructor (iv) Instructor – Student on the following tables: Student Fname | Lname Susan | Yao Ramesh | Shah Johnny | Kohler Barbara | Jones Amy | Ford Jimmy | Wang Ernest | Gilbert Instructor Fname | Lname John | Smith Ricardo | Browne Susan | Mao Francis | Johnson Ramesh | Shah	04	L3	CO2
	b.	Consider the following relational database schema and write the queries in relational algebra expressions: EMP(Eno, Ename, Salary, Address, Phone, DNo) DEPT(DNo, Dname, DLoc, MgrEno) DEPENDENT(Eno, Dep_Name, Drelation, Dage) (i) List all the employees who reside in 'Belagavi'. (ii) List all the employees who earn salary between 30000 and 40000 (iii) List all the employees who work for the 'Sales' department (iv) List all the employees who have at least one daughter (v) List the department names along with the names of the managers	10	L3	CO2

	c.	Consider the two tables T_1 and T_2 shown below: T_1 P | Q | R 10 | a | 5 15 | b | 8 25 | a | 6 T_2 A | B | C 10 | b | 6 25 | c | 3 10 | b | 5 Show the results of the following operations: (i) $T_1 \bowtie_{T_1.P=T_2.A} T_2$ (ii) $T_1 \bowtie_{T_1.Q=T_2.B} T_2$ (iii) $T_1 \bowtie_{(T_1.P=T_2.A \text{ AND } T_1.R=T_2.C)} T_2$	06	L3	CO2
		Module – 3			
Q.5	a.	Discuss the informal design guidelines for relation schema design.	08	L2	CO4
	b.	Define 1NF, 2NF, and 3NF with examples.	06	L2	CO4
	c.	Write the syntax for INSERT, UPDATE and DELETE statements in SQL and explain with suitable examples.	06	L2	CO3
		OR			
Q.6	a.	Discuss insertion, deletion and modification anomalies. Why are they considered bad? Illustrate with examples.	10	L2	CO3
	b.	Illustrate the following with suitable examples: (i) Datatypes in SQL (ii) Substring Pattern Matching in SQL.	10	L2	CO3
		Module – 4			
Q.7	a.	Consider the following relations: Student(Snum, Sname, Branch, level, age) Class(Cname, meet_at, room, fid) Enrolled(Snum, Cname) Faculty(fid, fname, deptid) Write the following queries in SQL. No duplicates should be printed in any of the answers. (i) Find the names of all Juniors (level = JR) who are enrolled in a class taught by I. Teach. (ii) Find the names of all classes that either meet in room R128 or have five or more students enrolled. (iii) For all levels except JR, print the level and rthe average age of students for that level. (iv) For each faculty member that has taught classes only in room R128, print the faculty member's name and the total number of classes she or he has taught. (v) Find the names of students not enrolled in any class.	10	L3	CO3
	b.	What do understand by correlated Nested Queries in SQL? Explain with suitable example.	04	L2	CO3
	c.	Discuss the ACID properties of a database transaction.	06	L2	CO4
		OR			
Q.8	a.	What are the views in SQL? Explain with examples.	04	L3	CO5
	b.	In SQL, write the usage of GROUP BY and HAVING clauses with suitable examples.	06	L2	CO3
	c.	Discuss the types of problems that may encounter with transactions that run concurrently.	10	L2	CO5

		Module – 5			
Q.9	**a.**	What is the two phase locking protocol? How does it Guarantee serializability.	**06**	**L2**	**CO5**
	b.	Describe the wait-die and wound-wait protocols for deadlock prevention.	**08**	**L2**	**CO5**
	c.	List and explain the four major categories of NOSQL system.	**06**	**L2**	**CO3**
		OR			
Q.10	**a.**	What is Multiple Granularity locking? How is it implemented using intension locks? Explain.	**10**	**L2**	**CO5**
	b.	Discuss the following MongoDB CRUD operations with their formats: (i) Insert (ii) Delete (iii) Read	**06**	**L2**	**CO4**
	c.	Briefly discuss about Neo4j data model.	**04**	**L2**	**CO4**

* * * * *

BCS403-database Management System-Scheme of Evaluation and Solution

1.a.**Database Definition (2 marks):**

A database is a shared collection of logically related data and its description, designed to meet the information needs of an organization.

Components (8 marks):

.a) Hardware: Computer system, storage devices (disk drives, tapes), I/O devices b) Software: DBMS software, operating system, application programs c) Data: Both operational data and the database description/metadata d) Procedures: Instructions and rules for database design and use e) People: Database administrators, designers, end users, and programmers f) Modules: DDL for data definition, DML for data manipulation, and Database Manager for file handling g) Database Manager: Controls data access, security, and integrity h) Data Dictionary: Stores metadata about database structure and constraints

1.b Three Schema Architecture:

- Definition (1 mark)
- Each schema description (3 marks - 1 mark each)
- Data independence types (2 marks - 1 mark each)

SOLUTION:

1. Definition (1 mark): Three-schema architecture is a framework that helps achieve database system properties of data independence and data abstraction using three levels of data description.
2. Schema Levels (3 marks):

a) External Schema (View Level):

- Describes database part visible to user group
- Multiple user views possible
- Hides complexity of logical structure

b) Conceptual Schema (Logical Level):

- Describes structure of whole database
- Includes logical relationships between data
- Independent of storage details

c) Internal Schema (Physical Level):

- Describes physical storage structure
- Handles data storage and file organization
- Deals with access paths and storage methods

3. Data Independence (2 marks):

a) Logical Data Independence:

- Ability to change conceptual schema without changing external views

b) Physical Data Independence:

- Ability to change internal schema without affecting conceptual schema

1.c.Difference between Logical and Physical Data Independence

1. Key differences (4 marks - 1 mark each point)

SOLUTION IN TABULAR FORMAT:

Logical Data Independence (3 marks)	**Physical Data Independence (3 marks)**
Involves changes at conceptual level without affecting external schema	Involves changes at physical level without affecting conceptual schema
Deals with changes in logical structure (like adding/removing tables, relationships)	Deals with changes in physical storage methods and access paths
More difficult to achieve as it may involve restructuring of database	Easier to achieve as it only involves internal storage modifications
Protects external user views from changes in conceptual design	Protects database logical design from changes in physical storage

Example:

- Logical: Adding new columns to tables without affecting existing applications
- Physical: Changing from sequential to random file organization without impacting database logic

2.a.ER Diagram for Company Database:

1. Entities and Attributes (3 marks)
    - Correct entity sets with rectangles
    - Primary keys underlined
    - Proper attribute notation
2. Relationships (3 marks)
    - Correct relationship diamonds
    - Proper relationship names
    - Accurate cardinality ratios
3. Constraints (2 marks)
    - Foreign key constraints
    - Participation constraints
4. ER Diagram (2 marks)

Key Features of the ER Diagram:

1. Main Entities:
    - EMPLOYEE (SSN, name, address, salary, etc.)
    - DEPARTMENT (dno, dname, mgr_ssn, etc.)
    - PROJECT (pnumber, pname, location)
    - DEPENDENT (essn, name, relationship)
2. Relationships:
    - WORKS_ON (between EMPLOYEE and PROJECT)
    - MANAGES (between EMPLOYEE and DEPARTMENT)
    - CONTROLS (between DEPARTMENT and PROJECT)
    - HAS_DEPENDENT (between EMPLOYEE and DEPENDENT)
3. Cardinality Constraints:
    - One department has one manager (1:1)
    - One department can have many employees (1:N)

- One employee can work on multiple projects (M:N)
- One employee can have multiple dependents (1:N)

2.b.Notations of ER Diagram:

1. Basic Components (4 marks)
2. Relationship Types & Notations (3 marks)
3. Attribute Types (3 marks)

SOLUTION:

1. Basic Components (4 marks):

a) Entity (1 mark)

- Represented by Rectangle ▭
- Example: EMPLOYEE, STUDENT
- Strong Entity: Single rectangle
- Weak Entity: Double rectangle ⬚

b) Relationship (1 mark)

- Represented by Diamond ⍰
- Connects two or more entities
- Example: WORKS_ON, MANAGES

c) Attributes (1 mark)

- Represented by Oval ○
- Connected to entity or relationship
- Primary Key: Underlined

d) Connecting Lines (1 mark)

- Solid lines: Regular entities
- Dotted lines: Weak entities
- Lines connect entities to relationships

2. Relationship Types & Notations (3 marks):

a) Cardinality Ratios:

- One-to-One (1:1) → ——||——||——
- One-to-Many (1:N) → ——||——<——
- Many-to-Many (M:N) → ——<——>——

b) Participation Constraints:

- Total: Double line (=)
- Partial: Single line (-)

c) Relationship Degrees:

- Binary: Two entities
- Ternary: Three entities
- n-ary: n entities

3. Attribute Types (3 marks):

a) Simple vs Composite:

- Simple: Single oval
- Composite: Ovals connected to oval

b) Single vs Multi-valued:

- Single: Single oval
- Multi-valued: Double oval

c) Derived vs Stored:

- Stored: Regular oval
- Derived: Dotted oval

3.a.Update operations and contraint violation :

- Types of updates (3 marks)
- Constraint violations (3 marks)
- Solutions/handling (2 marks)

SOLUTION:

a) Types of Update Operations (3 marks):

1. INSERT: Adding new tuples
2. DELETE: Removing existing tuples
3. MODIFY: Changing attribute values

b) Constraint Violations (3 marks):

1. Domain Constraints:

- Data type mismatch
- Value range violation

2. Key Constraints:

- Duplicate primary keys
- Null primary key values

3. Referential Integrity:

- Orphan records
- Invalid foreign keys

c) Handling Violations (2 marks):

1. RESTRICT/REJECT: Refuse invalid updates
2. CASCADE: Propagate changes automatically

3.b Relational algebra operators select ad project

- SELECT operator (3 marks)
- PROJECT operator (3 marks)

SOLUTION:

a) SELECT (σ) Operator (3 marks):

- Definition: Selects tuples based on condition
- Notation: σ<condition>(relation)
- Example: σsalary>50000(EMPLOYEE)
- Characteristics:
    - Horizontal partitioning
    - Returns matching rows

b) PROJECT (π) Operator (3 marks):

- Definition: Selects specific attributes
- Notation: π<attribute_list>(relation)
- Example: πname,salary(EMPLOYEE)
- Characteristics:
    - Vertical partitioning
    - Eliminates duplicates

3.c.characteristcis of relations-6 mark (1 Mark for each property)

1. Atomic Values:

- Each attribute value is indivisible

2. No Duplicate Tuples:

- Each row must be unique

3. No Order Among Tuples:

- Row order is insignificant

4. No Order Among Attributes:

- Column order is insignificant

5. Each Attribute Has Unique Name:

- Column names must be distinct

6. Each Cell Contains Single Value:

- No multi-valued attributes allowed

4.a.Problem: (each Subdivision carries 2.5 mark each)

(i) Student ∪ Instructor (Union)-(2.5M)

Fname	Lname
Susan	Yao
Ramesh	Shah

Johnny Kohler

Barbara Jones

Amy Ford

Jimmy Wang

Ernest Gilbert

John Smith

Ricardo Browne

Susan Mao

Francis Johnson

James Shah

(ii) Student ∩ Instructor (Intersection)-(2.5M)

Fname Lname

Susan Mao

James Shah

(iii) Student - Instructor (Difference)-(2.5M)

Fname Lname

Susan Yao

Ramesh Shah

Johnny Kohler

Barbara Jones

Amy Ford

Jimmy Wang

Ernest Gilbert

(iv) Instructor - Student (Difference)-(2.5M)

Fname Lname

John Smith

Ricardo Browne

Francis Johnson

4.b.Problem: (each Subdivision carries 2 mark each)

Given Schema:

- EMP(Eno, Ename, Salary, Address, Phone, DNo)
- DEPT(DNo, Dname, DLoc, MgrEno)
- DEPENDENT(Eno, Dep_Name, Drelation, Dage)

Solutions:

(i) List employees who reside in 'Belagavi'-(2M)

σAddress='Belagavi'(EMP)

(ii) List employees with salary between 30000 and 40000-(2M)

σSalary>=30000 ∧ Salary<=40000(EMP)

(iii) List employees who work for 'Sales' department-(2M)

πEMP.*(σDname='Sales'(EMP ⋈ DEPT))

(iv) List employees who have at least one daughter-(2M)

πEMP.Ename(σDrelation='daughter'(EMP ⋈ DEPENDENT))

(v) List employees with managers' names-(2M)

πE.Ename, M.Ename(ρE(EMP) ⋈ ρM(EMP ⋈ DEPT))

4.c Problem: (each Subdivision carries 2 mark each)

(i) T1 ⋈[T1.P=T2.A] T2 -(2M)

P/A	Q	R	B	C
10	a	5	b	6
10	a	5	h	5
25	a	6	c	3

(ii) T1 ⋈[T1.Q=T2.B] T2 -(2M)

P	Q/B	R	A	C
15	b	8	10	6

(iii) T1 ⋈[(T1.P=T2.A) AND (T1.R=T2.C)] T2 -(2M)

P/A Q R/C B

10 a 5 h

2. 5.a. INFORMAL DESIGN GUIDELINES FOR SCHEMA DESIGN (8 marks)

1. Ensure Semantics of Attributes -(2M)
    - Attributes should be clearly defined
    - Relationship between attributes should be meaningful
2. Reduce Redundancy -(2M)
    - Minimize duplicate information
    - Use foreign keys appropriately
3. Avoid Null Values -(2M)
    - Design tables to minimize null entries
    - Split tables if too many nulls exist
4. Ensure Data Integrity -(2M)
    - Define proper constraints
    - Maintain referential integrity

5.b. NORMAL FORMS WITH EXAMPLES (6 marks)

a) 1NF (First Normal Form) -(2M)

- Atomic values in each column
- No repeating groups Example: Before 1NF: Customer(ID, Name, Phone_Numbers) After 1NF: Customer(ID, Name), Phone(ID, Phone_Number)

b) 2NF (Second Normal Form) -(2M)

- Must be in 1NF
- No partial dependencies Example: Before 2NF: Order(Order_ID, Product_ID, Product_Name, Quantity) After 2NF: Order(Order_ID, Product_ID, Quantity), Product(Product_ID, Product_Name)

c) 3NF (Third Normal Form) -(2M)

- Must be in 2NF

- No transitive dependencies Example: Before 3NF: Employee(Emp_ID, Dept_ID, Dept_Location) After 3NF: Employee(Emp_ID, Dept_ID), Department(Dept_ID, Dept_Location)

5.c. SQL SYNTAX FOR DML STATEMENTS (6 marks)

a) INSERT Statement -(2M)

INSERT INTO table_name [(column_list)]

VALUES (value1, value2, ...);

b) UPDATE Statement -(2M)

UPDATE table_name

SET column1 = value1, column2 = value2

WHERE condition;

c) DELETE Statement -(2M)

DELETE FROM table_name

WHERE condition;

Variable Constraints:

- Must maintain referential integrity
- Check constraints before operations
- Handle null values appropriately
- Consider cascading effects

6.a. **a. Insertion, Deletion, and Modification Anomalies-10M**

- **Insertion Anomaly**-(2M)
- Occurs when you cannot insert data into a database due to the absence of other data. For example, if you cannot add a student without assigning them to a class.
- **Deletion Anomaly**-(2M)
- Occurs when deleting data inadvertently causes loss of additional data. For instance, deleting a class record might also remove related student data
- **Modification Anomaly:** -(2M)
- Occurs when changes to data require multiple updates. For example, changing a student's advisor requires updating multiple records.

- Examples--(4M)

6.b. **Datatypes in SQL- (5M)**

Examples include INT, VARCHAR, DATE, BOOLEAN, etc.

Substring Pattern Matching in SQL- (5M)

Use the LIKE operator, e.g., SELECT * FROM Students WHERE name LIKE '%John%'.

7.a. **SQL Queries-10M**

Given the relations:

- Student(Snum, Sname, Branch, level, age)
- Class(Cname, meet_at, room, fid)
- Enrolled(Snum, Cname)
- Faculty(fid, fname, deptid)

(i) **Find names of all Juniors enrolled in class taught by I. Teach:** -(2M)

```
SELECT DISTINCT S.Sname
FROM Student S
JOIN Enrolled E ON S.Snum = E.Snum
JOIN Class C ON E.Cname = C.Cname
JOIN Faculty F ON C.fid = F.fid
WHERE S.level = 'JR' AND F.fname = 'I. Teach';
```

(ii) **Find names of all classes meeting in room R128 or have five or more students enrolled:** -(2M)

```
SELECT DISTINCT C.Cname
FROM Class C
WHERE C.room = 'R128'
UNION
SELECT C.Cname
FROM Enrolled E
JOIN Class C ON E.Cname = C.Cname
```

```
GROUP BY C.Cname
HAVING COUNT(E.Snum) >= 5;
```

(iii) **For all levels except JR, print level and average age of students for that level:** -(2M)

```
SELECT S.level, AVG(S.age) AS average_age
FROM Student S
WHERE S.level <> 'JR'
GROUP BY S.level;
```

(iv) **For each faculty teaching only in room R128, print name and total classes taught:** -(2M)

```
SELECT F.fname, COUNT(DISTINCT C.Cname) AS total_classes
FROM Faculty F
JOIN Class C ON F.fid = C.fid
WHERE C.room = 'R128'
GROUP BY F.fname;
```

(v) **Find names of students not enrolled in any class:** -(2M)

```
SELECT S.Sname
FROM Student S
LEFT JOIN Enrolled E ON S.Snum = E.Snum
WHERE E.Cname IS NULL;
```

7.b.Correlated Nested Queries(2M+2M)

Correlated nested queries are subqueries that depend on the outer query for their values. They are executed once for each row processed by the outer query. Example:

```
SELECT S1.Sname
FROM Student S1
WHERE S1.age > (SELECT AVG(S2.age) FROM Student S2 WHERE S2.branch = S1.branch);
```

7.c. ACID Properties(6M)

Atomicity: -(2M)

Ensures that all operations within a transaction are completed; otherwise, none are.

Consistency: -(1M)

Ensures the database remains in a consistent state before and after the transaction.

Isolation: -(1M)

Ensures that concurrent transactions do not affect each other.

Durability: -(2M)

Ensures that once a transaction is committed, it will remain so, even in the event of a system failure.

8. **a. Views in SQL-(2+24M)**

Views are virtual tables in SQL that are based on the result of a SELECT query. They do not store data themselves but display data stored in other tables.**Example:**

```
CREATE VIEW StudentView AS

SELECT Sname, Branch

FROM Student

WHERE level = 'JR';
```

8.b. **Usage of GROUP BY and HAVING Clauses(3+3M)**

The GROUP BY clause is used to arrange identical data into groups.
The HAVING clause is used to filter records that work on aggregated data.

Example:

```
SELECT Branch, COUNT(*)

FROM Student

GROUP BY Branch

HAVING COUNT(*) > 10;
```

8.c. **Problems with Concurrent Transactions(10M)**

Concurrent transactions can lead to several issues:

Lost Update: -(2M)

When two transactions update the same data, and one update is lost.

Dirty Read: -(2M)

When a transaction reads data that has been modified by another transaction but not yet committed.

Non-Repeatable Read: -(2M)

When a transaction reads the same row twice and gets different data each time.

Phantom Read: -(2M)

When a transaction reads new rows added by another transaction.

Example for each-(2M)

9. a. Two-Phase Locking Protocol-(6M)

The Two-Phase Locking Protocol ensures serializability by dividing the execution of a transaction into two phases:

Growing Phase: -(3M)

Locks can be acquired but not released.

Shrinking Phase: -(3M)

Locks can be released but not acquired.

This protocol prevents cycles in the wait-for graph, ensuring serializability.

b. Wait-Die and Wound-Wait Protocols-(8M)

Wait-Die Protocol: -(4M)

Older transactions may wait for younger ones, but younger transactions requesting locks held by older ones are aborted ("die").

Wound-Wait Protocol: -(4M)

Older transactions requesting locks held by younger ones preempt the younger ones ("wound"), forcing them to abort.

These protocols help prevent deadlocks by ensuring a consistent ordering of transactions.

c. Four Major Categories of NoSQL Systems--(6M)

Key-Value Stores: (2M)

Simple storage systems using a key-value pair (e.g., Redis).

Document Stores: (2M)

Store data in document formats like JSON (e.g., MongoDB).

Column-Family Stores: (1M)

Store data in columns rather than rows (e.g., Cassandra).

Graph Databases: (1M)

Designed for data with complex relationships (e.g., Neo4j).

10.a. **a. Multiple Granularity Locking-10M**

Multiple Granularity Locking (2M)

It allows transactions to lock resources at various levels of granularity (e.g., database, table, row). It uses intention locks to indicate a transaction's intention to acquire locks at a finer granularity.**Example:**

Intention Shared (IS): (4M)

Intends to acquire shared locks.

Intention Exclusive (IX): (4M)

Intends to acquire exclusive locks.

10.b. **MongoDB CRUD Operations- (6M)**

i. Insert: - (2M)

```
db.collection.insertOne({ name: "Alice", age: 25 });
```

ii. Delete: - (2M)

```
db.collection.deleteOne({ name: "Alice" });
```

iii. Read: - (2M)

```
db.collection.find({ age: { $gt: 20 } });
```

10.c. A Neo4j model - (2M)

It is a representation of data in a **graph database format**, where data is stored as nodes, relationships, and properties rather than traditional tables and columns. This structure is highly effective for representing connected data, such as social networks, recommendation systems, and hierarchical structures, making Neo4j popular for these applications.

Core Elements of a Neo4j Model- (2M)

Nodes: The primary entities, like "Person," "Product," or "City." Each node can have labels that describe its role in the model (e.g., :User, :Product) and can also have various properties (like name, age, or creation date).

1. **Relationships**: Connect two nodes and represent their interactions or dependencies (e.g., FRIENDS_WITH, PURCHASED, LOCATED_IN). Relationships also have directions and can have properties (like relationship length, timestamp, etc.).
2. **Properties**: Attributes or values assigned to nodes or relationships, such as age, price, timestamp, or weight.

3. **Labels**: Categories that can be applied to nodes to group them (like :Person, :City). This helps simplify data queries.