Creating)

PART- A (10 x 2 = 20 Marks)

Q. No	Questions	Marks	CO	BL
1	What is a foreign key? Give example.	2	CO1	L1
2	Consider the relation $\partial(\alpha, \beta)$. Relation ∂ is in 1NF. Is relation ∂ in 2NF? Outline.	2	CO1	L1
3	Define a distributed database management system.	2	CO2	L1
4	Outline the need for concurrency control.	2	CO2	L1
5	What is MongoDB?	2	CO3	L1
6	Name any two SQL databases.	2	CO3	L1
7	What is JSON?	2	CO4	L1
8	Outline a collection in MongoDB.	2	CO4	L1
9	Define a graph database.	2	CO5	L1
10	What is CQL?	2	CO5	L1

PART- B (5 x 13 = 65 Marks)

Q. No	Questions	Marks	CO	
11 (a)	Consider the relation SPJ (SUPPLIER_PART_PROJECT) presented below:			
	SPJ **SNO** **PNO** **JNO** S101 P101 J101 S101 P102 J101 S102 P101 J102 S102 P102 J102 S103 P101 J103			
	A supplier can supply many parts. The same part can be supplied by may suppliers. Parts are used in projects. Decompose relation SPJ into relations SP (SNO, PNO), PJ (PNO, JNO) and SJ (SNO, JNO). Perform the following: i. Equijoin between relations SP and PJ. ii. Equijoin between relations SP and SJ. iii. Equijoin between relations SJ and PJ. Outline what you can infer from the resultant relations.	**13**	CO1	L3
	OR			
11 (b) (i)	Consider the relation schema R (A, B, C, D, E, F) with primary key AB. Relation R is in 1NF. Present a set of functional dependencies to justify relation R is not in 2NF and not in 3NF.	7	CO1	L3
(ii)	When is a relation R said to be in BCNF? Prove that any relation schema with two attributes is in BCNF.	6	CO1	L3
12 (a)	Present a distributed database design for a chain of "Volkswagen Service Centers". State the functional requirements you are considering.	**13**	CO2	L3
	OR			
12 (b)	Consider the relations presented below:	**13**	CO2	L3

	SUPPLIER (**SCODE**, SNAME) (Stored @site A) PART (**PCODE**, PNAME, COLOR) (Stored @site B) COLOR is a single valued attribute. SUPPLIES (**SCODE**, **PCODE**) (Stored @site C) The primary key of each relation is underlined. A Supplier can supply 1 or more Part's. The same Part can be supplied by more than one Supplier. Write relational algebra expressions (step by step) to list the details of Suppliers who supply BLUE Color Parts (@site D).			
13 (a)	What is a NoSQL database? Outline the features of any two types of NoSQL databases.	**13**	CO3	L1
	OR			
13 (b)	What is HIVE? Outline the features of HIVE.	**13**	CO2	L1
14 (a)	What is a polymorphic schema in MongoDB? Outline with an example.	**13**	CO4	L3
	OR			
14 (b)	Illustrate CRUD Operations in MongoDB for the relational schema presented in Question No. 12.b.	**13**	CO4	L3
15 (a)	Outline data modeling with graphs with an example.	**13**	CO5	L2
	OR			
15 (b)	Outline the CQL clauses and CQL functions with an example.	**13**	CO5	L2

PART- C (1 x 15 = 15 Marks)

Q. No	Questions	Marks	CO	BL
16.	A Company is organized into departments. Each department has employees working in it. The attributes of department include department number and department name. The attributes of employee include employee number, employee name, date of birth, gender, date of joining, designation and basic pay. Each department has a manager managing it. There are also supervisors in each department who supervise a set of employees. Each department			

	controls a number of projects. The attributes of project include project code and project name. A project is controlled only by one department. An employee can work in any number of distinct projects on a day. The date an employee worked, in time and out time has to be kept track. The company also keeps track of the dependents of each employee. The attributes of dependent include dependent name, date of birth, gender and relationship with the employee.			
(i)	Model an entity relationship diagram for the above scenario.	7	CO6	L6
(ii)	Map the entity relationship diagram you have modeled to relations.	8	CO6	L6

Here are the answers to the given questions:

1. **What is a foreign key? Give an example.**
    a. A foreign key is an attribute in one table that establishes a relationship with the primary key of another table.
    b. **Example:**
```
CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50)
);

CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    EmpName VARCHAR(50),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);
```

2. **Consider the relation ∂(α, β). Relation ∂ is in 1NF. Is relation ∂ in 2NF? Outline.**
    a. A relation is in **2NF** if it is in **1NF** and **has no partial dependencies** (i.e., no non-prime attribute should depend on a part of a composite primary key).
    b. If **α, β** form a composite key, and there is an attribute dependent only on **α** or **β**, then **∂ is not in 2NF**.
    c. To achieve 2NF, we decompose the relation so that all non-key attributes fully depend on the entire primary key.
3. **Define a distributed database management system.**

a. A **Distributed Database Management System (DDBMS)** is a system that manages a database distributed across multiple locations.
b. It ensures data consistency, transparency, and query execution across different sites.

4. **Outline the need for concurrency control.**
    a. Concurrency control is required to handle **simultaneous transactions** and ensure **data consistency**.
    b. It prevents issues like **dirty reads, lost updates, and uncommitted data problems**.
5. **What is MongoDB?**
    a. **MongoDB** is a **NoSQL database** that stores data in **BSON (Binary JSON) format**.
    b. It is a **document-oriented** database used for **scalability, flexibility, and high performance**.
6. **Name any two SQL databases.**
    a. **MySQL**
    b. **PostgreSQL**
7. **What is JSON?**
    a. **JSON (JavaScript Object Notation)** is a lightweight data format used for data exchange.
    b. **Example:**
```
{
  "name": "John",
  "age": 25,
  "city": "New York"
}
```

8. **Outline a collection in MongoDB.**
    a. A **collection** in MongoDB is a group of **documents** (similar to tables in SQL).
    b. Each document is stored in **BSON format** and has a flexible schema.
9. **Define a graph database.**
    a. A **graph database** stores data using **nodes, edges, and properties**.
    b. It is optimized for **relationship-based queries**.
    c. **Example:** Neo4j.
10. **What is CQL?**

- **CQL (Cassandra Query Language)** is the query language used in **Apache Cassandra** to interact with its database.
- **Example Query:**
```
CREATE TABLE users (
    id UUID PRIMARY KEY,
```

```
    name TEXT,
    age INT
);
```

PART - B (5 × 7 = 35 Marks)

11. (a) Explain the characteristics of a Database Management System (DBMS).

A **DBMS** has the following characteristics:

1. **Data Abstraction** – Hides complexity of data representation.
2. **Data Independence** – Changes in schema do not affect applications.
3. **Data Security** – Provides authentication and authorization.
4. **ACID Properties** – Ensures **Atomicity, Consistency, Isolation, and Durability**.
5. **Concurrency Control** – Manages multiple transactions simultaneously.
6. **Data Integrity** – Enforces constraints like **Primary Key and Foreign Key**.
7. **Backup and Recovery** – Prevents data loss in case of failure.

💡 **Example:** MySQL, Oracle, PostgreSQL, MongoDB are DBMSs.

(OR)

(b) Discuss the different types of data models in DBMS.

There are four major types of data models:

1. **Hierarchical Model** – Organizes data in a **tree structure**.
    a. **Example:** IBM's IMS DB
2. **Network Model** – Represents data as **graphs**, allowing multiple parent-child relationships.
    a. **Example:** CODASYL DBMS
3. **Relational Model** – Uses **tables (relations)** with rows and columns.
    a. **Example:** MySQL, PostgreSQL
4. **Object-Oriented Model** – Stores data as **objects** (like in OOP).
    a. **Example:** db4o, ObjectDB

12. (a) Explain functional dependencies with examples.

- A **functional dependency (FD)** exists when one attribute uniquely determines another.
- **Notation:** $X \rightarrow Y$ (X determines Y).
- **Example:** In a STUDENT table, `StudentID → StudentName`
  `StudentID, CourseID → Grade`

- **Types of FD:**
  - **Trivial FD** – $A,B \rightarrow A$ ($A, B \rightarrow A$ (already contained).
  - **Non-Trivial FD** – $A \rightarrow B$ ($A \rightarrow B$ (B is not in A).
  - **Multivalued FD** – $A \twoheadrightarrow B$ ($A \twoheadrightarrow B$ (A determines multiple B values).

(OR)

(b) Describe the different types of normalization in DBMS.

Normalization removes redundancy and improves database efficiency.

1. **1NF (First Normal Form)** – Removes **duplicate columns** and ensures **atomicity**.
2. **2NF (Second Normal Form)** – Removes **partial dependencies** (only applies if composite keys exist).
3. **3NF (Third Normal Form)** – Removes **transitive dependencies**.
4. **BCNF (Boyce-Codd Normal Form)** – Stronger than 3NF, ensures no partial dependencies exist.

💡 **Example:**

1NF:

```
EmpID | Name | Skills
101   | A    | Java, C++
```

👉 Convert into two tables:

```
EmpID | Name
101   | A
```

```
EmpID | Skill
101   | Java
101   | C++
```

Now it is in **1NF**.

13. (a) Explain the ACID properties of transactions.

ACID properties ensure reliable transactions:

1. **Atomicity** – A transaction is **all-or-nothing**.
2. **Consistency** – Transactions keep the database in a **valid state**.
3. **Isolation** – Transactions do not **interfere** with each other.
4. **Durability** – Committed transactions are **permanent**, even in crashes.

Example:

```
START TRANSACTION;
UPDATE Accounts SET Balance = Balance - 500 WHERE AccountID = 1;
UPDATE Accounts SET Balance = Balance + 500 WHERE AccountID = 2;
COMMIT;
```

If any step fails, **ROLLBACK** ensures data remains consistent.

(OR)

(b) Describe the different types of concurrency control techniques.

Concurrency control ensures correct execution of simultaneous transactions.

1. **Lock-Based Protocols**
    a. **Two-Phase Locking (2PL)** – Divides transaction into **growing** and **shrinking** phases.
    b. **Shared/Exclusive Locks** – Read (shared) and write (exclusive) locks.
2. **Timestamp-Based Protocols**
    a. Transactions have timestamps to ensure order.
3. **Optimistic Concurrency Control**

a. Checks conflicts at commit time instead of locking.

14. (a) Explain the data types supported by MongoDB with examples.

MongoDB supports:

1. **String** – `"name": "Alice"`
2. **Number (Integer, Double, Long, Decimal)** – `"age": 25`
3. **Boolean** – `"active": true`
4. **Array** – `"skills": ["Java", "Python"]`
5. **Object (Embedded Document)** – `"address": { "city": "NY", "zip": "10001" }`
6. **Date** – `"created_at": ISODate("2025-03-14T10:00:00Z")`
7. **Binary Data** – Stores images, PDFs, etc.

(OR)

(b) Compare SQL and NoSQL databases.

Feature	SQL (Relational)	NoSQL (Non-Relational)
Schema	Fixed (Predefined)	Dynamic (Flexible)
Scalability	Vertical (Scaling Up)	Horizontal (Scaling Out)
Data Storage	Tables (Rows & Columns)	Key-Value, Document, Graph
Examples	MySQL, PostgreSQL	MongoDB, Cassandra
Best for	Structured Data	Unstructured Data

15. (a) Explain CRUD operations in MongoDB with examples.

CRUD = **Create, Read, Update, Delete**

1. **Create (Insert)** `db.students.insertOne({ "name": "John", "age": 20 });`
2. **Read (Find)** `db.students.find({ "age": 20 });`

3. **Update** `db.students.updateOne({ "name": "John" }, { $set: { "age": 21 } });`

4. **Delete** `db.students.deleteOne({ "name": "John" });`

(OR)

(b) Write a short note on the following:

1. **Key-Value Store**
    a. Stores **key-value pairs**, like a dictionary.
    b. **Example:** Redis, DynamoDB.
2. **Column-Family Store**
    a. Stores **data in columns** instead of rows.
    b. **Example:** Apache Cassandra.
3. **Document Store**
    a. Stores **JSON-like documents**.
    b. **Example:** MongoDB.
4. **Graph Database**
    a. Uses **nodes and edges** to represent relationships.
    b. **Example:** Neo4j.

PART - C (1 × 15 = 15 Marks)

16. (a) Explain how distributed databases are managed.

1. **Model an Entity-Relationship (ER) Diagram** for a company system where:
    a. A company has departments.
    b. Each department has a department number and name.
    c. Each department has employees with attributes like employee ID, name, DOB, gender, joining date, designation, and basic pay.
    d. Each department has a **manager**.
    e. There are **supervisors** in each department supervising employees.
    f. Departments control **projects** (one department per project).
    g. Employees can work on multiple projects.
    h. The company tracks the **time worked** by employees on projects.

    i. The company tracks **dependents** of employees with attributes like name, DOB, gender, and relationship.
2. **Map the ER Diagram to relations** by converting entities, relationships, and attributes into relational tables.

(i) ER Diagram for the given scenario

Your ER diagram should include:

- **Entities:** Department, Employee, Project, Dependent.
- **Relationships:**
    - **Manages** (Department → Employee as Manager) [1:1]
    - **Supervises** (Supervisor → Employee) [1:M]
    - **Works_on** (Employee ↔ Project) [M:N]
    - **Has_Dependent** (Employee → Dependent) [1:M]
    - **Controls** (Department → Project) [1:M]
    - **Tracks_Time** (Employee ↔ Project with attributes Date, In-Time, Out-Time) [M:N]

(ii) Mapping ER Diagram to Relations

Based on the ER model, create the following relations:

1. **Department (DeptID, DeptName, ManagerID)**
2. **Employee (EmpID, EmpName, DOB, Gender, JoinDate, Designation, BasicPay, DeptID, SupervisorID)**
3. **Project (ProjID, ProjName, DeptID)**
4. **Works_On (EmpID, ProjID, Date, InTime, OutTime)**
5. **Dependent (DepID, EmpID, DepName, DOB, Gender, Relationship)**