**Database Management Assignment:-**

**Section A: Introduction to SQL/NoSQL**

1. **You are working on a project where you need to store large amounts of structured and semi-structured data. Which type of database (SQL or NoSQL) would you choose and why? Explain with a practical example.**

If I were working with both structured and semi-structured data, I would opt for a **NoSQL database** like **MongoDB**. NoSQL is more flexible and does not require a rigid table structure like SQL, making it easier to handle data in varying formats.

For example, in a **smart home project**, structured data like **device IDs** and **timestamps** can be combined with semi-structured data such as **sensor readings (temperature, humidity)**. NoSQL efficiently stores and retrieves both types of data, making it ideal for such applications.

1. **A company wants to migrate from a relational database to a NoSQL database for better scalability. What challenges might they face? Discuss with an example.**

Shifting from SQL to NoSQL can bring a few challenges:  
- Data model has to be redesigned completely.  
- The team needs to learn a new query language and database structure.  
- Joins are not as straightforward in NoSQL.  
- Consistency might be an issue since NoSQL usually focuses more on speed and scalability than strict accuracy.  
For instance, an e-commerce company might struggle to organize customer-order relationships efficiently in NoSQL.

**Section B: Advantages and Disadvantages of SQL/NoSQL**  
**3. You are designing an e-commerce website's database. Explain the advantages and disadvantages of using SQL vs. NoSQL in this scenario.**  
SQL is great for consistency, complex queries, and strict relationships between orders, payments, and customers.  
NoSQL works better for scalability and flexibility for handling user reviews, dynamic product listings, etc.  
A combination of both (hybrid approach) is often ideal.

**4. A banking system requires high consistency and ACID compliance. Which database system (SQL or NoSQL) would you recommend? Justify your answer with a real-world use case.**

SQL is better because banking systems need ACID compliance, high accuracy, and consistency.  
For example, a money transfer should either complete fully or not at all, which is ensured by SQL.

**Section C: Managing Databases  
5. You are a database administrator and need to perform routine maintenance on a production database. Describe at least three essential database management tasks you would perform.**

- Regular backups  
- Query performance monitoring  
- Index management  
- Checking for unauthorized access

**6. An online streaming service needs to optimize its database performance. What strategies can be used for effective database management in this case?**

- Use caching systems (e.g., Redis)  
- Index frequently used columns  
- Database sharding  
- Data archiving  
- Replication for read scaling

**Section D: Identifying System Databases in SQL Server**  
**7. List and describe the system databases in SQL Server. Provide one practical use case for each system database.**  
**- master:** stores configuration and server-level info  
**- model:** template for new databases  
**- msdb:** handles jobs, backups  
**- tempdb:** temporary data for sorting, joins, etc.

**8. You have accidentally deleted a user database in SQL Server. Which system database would you use to recover it, and how?**

Use msdb to locate the latest backup.  
Restore using the backup file and apply transaction logs if necessary.

**Section E: Normalization Forms (1NF, 2NF, 3NF, BCNF)**

**9. Given the following unnormalized table:**

| **OrderID** | **CustomerName** | **Product** | **Quantity** | **SupplierName** | **SupplierContact** |
| --- | --- | --- | --- | --- | --- |
| 101 | John Doe | Laptop | 1 | ABC Ltd. | 1234567890 |
| 102 | Jane Smith | Phone | 2 | XYZ Inc. | 9876543210 |

**Convert it to 1NF, 2NF, and 3NF with proper explanations**.

1NF - Atomic values, no repetition.  
2NF - Split:  
 Orders(OrderID, CustomerName)  
 OrderDetails(OrderID, Product, Quantity, SupplierName)  
3NF - Create Suppliers(SupplierName, SupplierContact) to remove transitive dependency.

1. **A company is facing redundancy issues in their database. How would applying BCNF help reduce redundancy? Explain with an example.**

It Removes dependency on non-primary keys.

**Example:**

**Before:** Course → Instructor → Room

**After:**

**InstructorRoom(Instructor, Room)**

**CourseInstructor(Course, Instructor)**

**End of Question Paper**