**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 need to store large amounts of structured data, I’d go with an SQL database like MySQL or PostgreSQL because they offer strong consistency and powerful query capabilities. But if my data includes semi-structured elements like JSON or XML, a NoSQL database such as MongoDB makes more sense since it allows flexible schema design and handles unstructured data better. **Example:** A hospital storing patient records in an SQL database for reliability but using NoSQL for unstructured medical notes and logs.

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.

Since NoSQL doesn't use traditional tables and relationships, data modeling must be adjusted.SQL databases use structured queries, while NoSQL databases require different approaches like JSON-based queries. SQL databases ensure ACID compliance, while some NoSQL databases prioritize scalability over consistency. An online store shifting from MySQL to MongoDB may struggle to maintain referential integrity across product listings and customer orders.

**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:

Great for handling relationships, strong consistency, and structured queries.

Can be hard to scale when traffic grows.

NoSQL:

More flexible schema, better scalability, and easier to handle big data.

Complex queries can be harder to manage, and ensuring consistency can be tricky.

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.

Definitely SQL, because banks need high consistency and ACID compliance to ensure transaction reliability. **Eg:** When transferring money between accounts, SQL ensures that either the full transaction happens or none of it does—preventing errors like duplicate deductions or missing credits.

**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:** Prevent data loss in case of system failures.

**Optimizing Indexes:** Keeps queries running fast and smooth.

**Monitoring Performance:** Checking logs and tuning queries to avoid slowdowns.

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

**Sharding:** Splitting data across multiple servers to balance load.

**Caching:** Storing frequently accessed data in-memory (e.g., Redis) to speed up response times.

**Load Balancing:** Distributing database requests efficiently across multiple instances.

**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 system-level info (e.g., login accounts, database configurations).

**msdb:** Manages scheduled tasks like backups and jobs.

**model:** Acts as a template for creating new databases.

**tempdb:** Handles temporary objects like session-specific tables and sorting operations.

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

The **msdb database** stores backup history, so I’d use it to restore the latest backup.

**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**

OrderID CustomerName Product Quantity SupplierName SupplierContact

101 John Doe Laptop 1 ABC Ltd. 1234567890

102 Jane Smith Phone 1 XYZ Inc. 9876543210

102 Jane Smith Phone 1 XYZ Inc. 9876543210

**2NF**

OrderID CustomerName Product Quantity

101 John Doe Laptop 1

102 Jane Smith Phone 1

102 Jane Smith Phone 1

Product SupplierName SupplierContact

Laptop ABC Ltd. 1234567890

Phone XYZ Inc. 9876543210

**3NF**

OrderID CustomerName Product Quantity

101 John Doe Laptop 1

102 Jane Smith Phone 1

102 Jane Smith Phone 1

Product SupplierID

Laptop 1

Phone 2

SupplierID SupplierName SupplierContact

1 ABC Ltd. 1234567890

2 XYZ Inc. 9876543210

StudentID StudentName Course

1 Alice DBMS

2 Bob DBMS

3 Charlie Networks

Course Instructor

DBMS Prof. X

Networks Prof. Y

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

BCNF ensures that every determinant is a candidate key, eliminating redundancy. **Example:** In a university database, if CourseID --> Professor and Professor --> Department, moving to BCNF separates professors into their own table so that department details don’t get repeated unnecessarily.

**End of Question Paper**