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 storing large amounts of structured and semi-structured data, NoSQL is preferred for scalability and flexibility. Example: A social media platform storing user profiles (structured) and posts with images, videos, and comments (semi-structured) benefits from NoSQL databases like MongoDB.

2) 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.

Challenges in migrating from SQL to NoSQL include data model transformation, lack of ACID compliance in some NoSQL databases, and application-level changes. Example: An e-commerce company moving from MySQL to Cassandra may struggle with complex queries requiring joins, which are not natively supported.

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 advantageous for complex queries, consistency, and structured data, but lacks scalability. NoSQL offers high scalability and flexibility but lacks strong ACID compliance. Example: A product catalogue benefits from SQL for relational integrity, whereas user behaviour tracking benefits from NoSQL.

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.

A banking system requires SQL due to ACID compliance, ensuring transaction integrity. Example: A bank’s transaction system must ensure balance updates occur correctly, preventing issues like double-spending.

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.

Essential database management tasks:

* + Backup and recovery to prevent data loss
  + Index optimization to enhance query performance
  + Monitoring and tuning for efficient resource utilization

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

Strategies to optimize an online streaming service’s database performance:

* + Caching frequently accessed data
  + Sharding to distribute load across multiple servers
  + Indexing to improve search speeds

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 configuration, user logins (Use: Managing instance-level settings)
  + model: Template for new databases (Use: Setting default configurations for new databases)
  + msdb: Manages jobs, alerts, and backups (Use: Scheduling automated backups)
  + tempdb: Temporary storage for intermediate query results (Use: Storing temporary tables and indexes)

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

To recover a deleted user database, use msdb to restore from a backup using: RESTORE DATABASE [database\_name] FROM DISK = 'backup\_path'  
If no backup exists, third-party recovery tools may be required.

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

1. Unnormalized Table:

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

1. **1NF:** Remove repeating groups:

| **OrderID** | **CustomerName** | **Product** | **Quantity** | **SupplierID** | **SupplierName** | **SupplierContact** |
| --- | --- | --- | --- | --- | --- | --- |
| 101 | John Doe | Laptop | 1 | S1 | ABC Ltd. | 1234567890 |
| 102 | Jane Smith | Phone | 2 | S2 | XYZ Inc. | 9876543210 |

1. **2NF:** Remove partial dependencies by creating separate supplier table: Orders Table:

| **OrderID** | **CustomerName** | **Product** | **Quantity** | **SupplierID** |
| --- | --- | --- | --- | --- |
| 101 | John Doe | Laptop | 1 | S1 |
| 102 | Jane Smith | Phone | 2 | S2 |

1. Suppliers Table:

| **SupplierID** | **SupplierName** | **SupplierContact** |
| --- | --- | --- |
| S1 | ABC Ltd. | 1234567890 |
| S2 | XYZ Inc. | 9876543210 |

1. **3NF:** Remove transitive dependency by creating a customer table: Customers Table:

| **CustomerID** | **CustomerName** |
| --- | --- |
| C1 | John Doe |
| C2 | Jane Smith |

1. Orders Table:

| **OrderID** | **CustomerID** | **Product** | **Quantity** | **SupplierID** |
| --- | --- | --- | --- | --- |
| 101 | C1 | Laptop | 1 | S1 |
| 102 | C2 | Phone | 2 | S2 |

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

BCNF removes redundancy by ensuring every determinant is a candidate key. Example:

* If a university has a table storing (StudentID, Course, Instructor), and one instructor teaches only one course, Instructor determines Course, leading to redundancy.
* Splitting into two tables (StudentID, Course) and (Course, Instructor) ensures BCNF compliance, eliminating redundancy.