**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.

**Ans:** When storing large amounts of structured and semi-structured data, a NoSQL database would be preferable due to its scalability and flexibility. NoSQL databases, such as MongoDB, allow for horizontal scaling, which is beneficial for handling large datasets. For example, an e-commerce platform managing user reviews, product details, and customer interactions benefits from NoSQL's schema-less structure, enabling seamless data integration.

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.

**Ans:** Migrating from a relational database to a NoSQL database poses several challenges, including data consistency, complex queries, and compatibility with existing applications. For instance, a financial institution transitioning from MySQL to Cassandra might struggle with ACID compliance, requiring careful planning to ensure transactional integrity.

**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.

Ans: In an e-commerce database, SQL databases provide strong consistency and structured query capabilities, making them ideal for transactional data like orders and payments. However, they may struggle with scaling. Conversely, NoSQL databases offer flexibility and high availability, useful for managing user-generated content and recommendations but can introduce data consistency challenges.

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.

Ans: A banking system requires SQL due to its need for high consistency and ACID compliance. For example, PostgreSQL ensures that transactions, such as money transfers, occur reliably and securely, preventing anomalies like double-spending.

**Use Case:** Consider an online banking system where a customer transfers money from one account to another. The database must ensure that if the amount is deducted from the sender’s account, it is credited to the receiver’s account simultaneously. If any failure occurs, the transaction must be rolled back to maintain data integrity. SQL databases, such as PostgreSQL or MySQL, provide strong ACID properties to ensure this level of consistency and reliability.

**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.

Ans:

* **Backup and Recovery:** Ensuring regular backups to prevent data loss.
* **Index Optimization:** Improving query performance by managing indexes effectively.
* **Monitoring and Performance Tuning:** Analyzing database metrics and optimizing slow queries.

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

Ans:

* **Database Sharding:** Distributing data across multiple servers.
* **Caching Mechanisms:** Using Redis or Memcached to reduce database load.
* **Load Balancing:** Distributing traffic efficiently to avoid bottlenecks.

**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-wide configuration data (e.g., login accounts).

Use Case: Restoring user access permissions after a system failure.

* **model:** Serves as a template for new databases.

Example: Creating a new database with predefined settings for consistency across deployments.

* **msdb:** Manages SQL Server Agent jobs and alerts.

Example*:* Automating database backup jobs to run at scheduled intervals.

* **tempdb:** Handles temporary storage for query processing.

Example: Storing temporary tables and intermediate query results to optimize performance.

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

Ans: To recover a deleted user database, use the **msdb** database, which contains backup history. A recovery process involves restoring from the last full backup using the RESTORE DATABASE command.

**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.

Ans:

1NF: Ensure atomicity by separating repeating values into different rows.

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

2NF: Remove partial dependencies by creating separate tables for products and suppliers.

Orders Table:

|  |  |
| --- | --- |
| OrderID | CustomerName |
| OrderID | CustomerName |
| 101 | John Doe |

Products Table:

|  |  |  |
| --- | --- | --- |
| ProductID | Product | Quantity |
| 1 | Laptop | 1 |
| 2 | Phone | 2 |

Suppliers Table:

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

3NF: Remove transitive dependencies by ensuring non-key attributes depend only on primary keys.

Here, SupplierContact is moved to the Suppliers table, ensuring only direct dependencies exist.

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

Ans: **BCNF:** BCNF eliminates redundancy by ensuring every determinant is a candidate key. For example, if a university database has a table:

|  |  |  |
| --- | --- | --- |
| StudentID | Course | Instructor |
| 1 | Math | Dr. Smith |
| 2 | Science | Dr. Johnson |

If an instructor only teaches one course, moving the instructor data to a separate table reduces redundancy, maintaining data integrity.

**End of Question Paper**