Overview:

A database is a structured collection of data that is organized and stored in a manner that allows for efficient retrieval, manipulation, and management.

It serves as a centralized repository for storing and managing data related to various aspects of an organization or application.

Database Management System (DBMS)

A DBMS is a software system that enables users to define, create, maintain, and manipulate databases.

It provides a set of tools and functionalities for managing data, enforcing data integrity, and facilitating data access.

DBMS Architecture

Three-Tier Architecture:

Presentation Tier : Interface for users to interact with the database.

Application Tier : Business logic and application processing.

Data Tier : Storage and retrieval of data.

Client-Server Architecture:

Client machines interact with a centralized database server.

Server handles data storage, retrieval, and processing.

Distributed Database Architecture:

Data is distributed across multiple locations or servers.

Provides scalability, fault tolerance, and improved performance.

Data Independence

Logical Data Independence:

Changes to the logical structure (schema) of the database do not affect the application programs accessing the data.

Physical Data Independence:

Changes to the physical storage of data do not affect the logical structure or application programs.

Integrity Constraints

Integrity constraints are rules that govern the validity and consistency of data stored in a database.

Examples include:

Entity Integrity :

Ensures that primary key values are unique and not null.

Referential Integrity :

Ensures that relationships between tables are maintained, and foreign key values reference existing primary key values.

Domain Integrity :

Enforces rules regarding the allowable values for attributes.

Check Constraints :

Specifies conditions that data must satisfy for insertion or modification.

Data Models

A data model is a conceptual representation of data structures and relationships.

It provides a framework for describing the structure, behaviour, and constraints of a database.

Relational Model

The relational model organizes data into tables (relations) consisting of rows (tuples) and columns (attributes).

Tables represent entities, and relationships between entities are represented by foreign keys.

Relational databases use SQL (Structured Query Language) for data manipulation and querying.

ER Model (Entity-Relationship Model)

The ER model is a graphical representation used to design and visualize database schemas.

It depicts entities as rectangles, attributes as ovals, and relationships as lines connecting entities.

Cardinality and participation constraints define the nature and degree of relationships between entities.

ER Diagram

An ER diagram is a visual representation of an ER model.

It illustrates the entities, attributes, and relationships in a database schema.

Helps in understanding the structure and semantics of the database.

Functional Dependencies and Normalization

Functional Dependencies

It describe the relationships between attributes in a relation.

If two attributes are functionally dependent, the value of one attribute uniquely determines the value of the other.

Example: In a relation R(A, B, C), if A uniquely determines B, it can be represented as A -> B.

Normalization:

Normalization is the process of organizing data in a database to reduce redundancy and dependency.

First Normal Form (1NF):

Eliminates repeating groups and ensures each attribute contains atomic values.

Second Normal Form (2NF):

All non-key attributes are fully functionally dependent on the primary key.

Third Normal Form (3NF):

Eliminates transitive dependencies, where non-key attributes depend on other non-key attributes.

Decomposition

It involves breaking a relation into smaller, more manageable relations to achieve higher normal forms.

It ensures that each relation satisfies a specific normal form and maintains data integrity.

Full Functional Dependency (FFD)

A functional dependency is full if removing any attribute from the determinant results in a loss of dependency.

Example:

A -> B is a full functional dependency if removing any attribute from A results in loss of dependency.

Transitive Dependency

It occurs when a non-key attribute depends on another non-key attribute.

It violates the third normal form and can lead to data redundancy.

Example:

In a relation R(A, B, C), if A -> B and B -> C, then A -> C is a transitive dependency.

Boyce-Codd Normal Form (BCNF)

BCNF is an extension of the third normal form (3NF).

It eliminates all non-trivial functional dependencies where the determinant is a superkey.

A relation is in BCNF if every determinant is a candidate key.

De-Normalization:

It is the process of intentionally adding redundancy to a database design for performance optimization.

It involves introducing redundancy to reduce the number of joins and improve query performance.

SQL Queries

DDL Statements

CREATE : Creates a new table, view, or index.

ALTER : Modifies the structure of an existing database object.

DROP : Deletes an existing database object.

DML Statements

INSERT : Adds new rows of data into a table.

UPDATE : Modifies existing data in a table.

DELETE : Removes rows from a table.

Simple Queries

SELECT : Retrieves data from one or more tables.

WHERE Clause : Filters rows based on specified conditions.

Compound WHERE Clause :

Combines multiple conditions using logical operators (AND, OR).

Joins

INNER JOIN :

Retrieves records that have matching values in both tables.

LEFT JOIN :

Retrieves all records from the left table and matching records from the right table.

RIGHT JOIN :

Retrieves all records from the right table and matching records from the left table.

FULL JOIN :

Retrieves all records when there is a match in either left or right table.

Sub-queries

Simple Sub-query :

A query nested within another query.

Correlated Sub-query :

Dependent on the outer query and executed for each row processed by the outer query.

Using IN, EXISTS, NOT EXISTS

IN :

Checks if a value matches any value in a list of values.

EXISTS :

Tests for the existence of any rows in a subquery.

NOT EXISTS:

Tests for the non-existence of any rows in a subquery.

DCL Statement

GRANT : Provides specific privileges to users or roles.

REVOKE : Revokes previously granted privileges.

Database Security

Introduction

It involves protecting the confidentiality, integrity, and availability of data stored in a database.

It encompasses various measures and mechanisms to prevent unauthorized access, data breaches, and malicious activities.

Threats

Unauthorized Access:

Unauthorized users gaining access to sensitive data or database resources.

Data Breaches:

Unauthorized disclosure of sensitive information.

SQL Injection :

Malicious SQL statements injected into input fields to manipulate or retrieve data.

Malware Attacks :

Viruses, worms, or other malicious software targeting databases.

Insider Threats :

Unauthorized actions by employees, contractors, or other trusted entities.

Counter Measures

Access Control :

Implement role-based access control (RBAC) to restrict access based on user roles and privileges.

Encryption :

Encrypt sensitive data at rest and in transit to protect against unauthorized access.

Authentication :

Use strong authentication mechanisms such as multi-factor authentication (MFA) to verify user identities.

Auditing and Logging :

Monitor database activities, track user actions, and maintain audit logs for forensic analysis.

Patch Management :

Regularly update and patch database software and security vulnerabilities to mitigate potential risks.

Database Activity Monitoring (DAM) :

Implement tools and systems to monitor and analyze database activities in real-time.

Backup and Disaster Recovery :

Implement backup and recovery procedures to ensure data availability and continuity in case of security incidents or disasters.

Employee Training and Awareness :

Provide security awareness training to employees to educate them about security best practices and potential threats.

Control Structures

Conditional Statements

They allow for the execution of different code blocks based on specified conditions.

Common conditional statements include IF, ELSEIF, and ELSE.

Iterative Control

They allow for the repetition of code blocks until certain conditions are met.

Common iterative control structures include FOR loops, WHILE loops, and DO-WHILE loops.

Sequential Control Statements:

They are executed sequentially, one after another, without any condition or loop.

They are used for executing a series of instructions in a specific order.

Cursors

Cursors are database objects used to retrieve and manipulate data row by row.

They enable the traversal of result sets returned by SQL queries and allow for data manipulation operations.

Views

Views are virtual tables generated by queries and do not store data themselves.

They provide a way to present data in a customized format and can simplify complex queries by abstracting underlying data structures.

Package, Procedures, and Triggers

Procedures

Procedures are named PL/SQL blocks that perform specific tasks or operations.

They consist of a declaration section, an executable section, and an optional exception handling section.

Parts of Procedures

1. **Declaration Section:**

   Defines variables, constants, cursors, and other program elements.

2. **Executable Section :**

   Contains the actual logic or code to be executed.

3. **Exception Handling Section:**

   Handles errors or exceptions that occur during procedure execution.

Parameter Modes

IN :

Passes values from the calling program to the procedure.

OUT :

Returns values from the procedure to the calling program.

IN OUT :

Passes values from the calling program to the procedure and returns modified values to the calling program.

Advantages of Procedures

Encapsulation:

Procedures encapsulate complex logic, making it reusable and easier to manage.

Modularity:

Procedures break down large tasks into smaller, more manageable units.

Security:

Procedures can restrict access to sensitive data or operations.

Performance:

Procedures can improve performance by reducing network traffic and minimizing repeated code execution.

Triggers

Syntax for Creating Triggers:

```
CREATE OR REPLACE TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF} {INSERT | UPDATE | DELETE}
ON table_name
[FOR EACH ROW | FOR EACH STATEMENT]
DECLARE
   -- Declaration section
BEGIN
   -- Executable section
END;
```

Types of Triggers

BEFORE Triggers :

Fired before the triggering event occurs.

AFTER Triggers :

Fired after the triggering event occurs.

INSTEAD OF Triggers :

Fired instead of the triggering event.

Package

Package Specification :

Declares public elements (constants, types, variables, exceptions) visible to other programs.

Package Body :

Implements the declared elements and includes private elements (procedures, functions, variables) not visible outside the package.

Developing a Package

1**. Create Package Specification** :

Declare public elements.

2. **Create Package Body :**

Implement public and private elements.

3. **Compile Package :**

Compile both specification and body.

Bodiless Package

It only includes the package specification without a body.

It is useful for defining constants, types, or global variables without implementation details.

Advantages:

Encapsulation:

Packages encapsulate related elements, providing a namespace and logical grouping.

Modularity:

Packages break down complex systems into manageable units, promoting code reusability.

Performance:

Packages reduce network traffic by storing program units on the server and minimizing parsing overhead.

Transaction Management and Concurrency Control

Transaction Management

A transaction is a logical unit of work that consists of one or more database operations.

ACID properties (Atomicity, Consistency, Isolation, Durability) ensure data integrity and consistency.

Concurrency Control

Intro to Transactions:

Transactions enable multiple users to access and manipulate data concurrently without causing data corruption or inconsistency.

Properties of Transactions:

1. **Atomicity:**

Transactions are all-or-nothing; either all operations succeed, or none are applied.

2. **Consistency:**

Transactions bring the database from one consistent state to another.

3. **Isolation:**

Transactions are executed independently of each other.

4. **Durability :**

Changes made by committed transactions are permanent and survive system failures.

Serializability and Recoverability

Serializability :

Ensures that transactions can be executed serially without interfering with each other's execution.

Recoverability :

Ensures that changes made by committed transactions are not lost due to system failures.

Need for Concurrency Control

Concurrent access to data can lead to various problems, including lost updates, uncommitted data, and inconsistent analysis.

Locking Techniques

Shared Locks:

Allow multiple transactions to read a resource but prevent any transactions from writing to it.

Exclusive Locks :

Prevent other transactions from reading or writing to a resource.

Two-Phase Locking (2PL):

Acquires all locks before performing any updates and releases all locks after the transaction is complete.

Deadlock Detection and Resolution :

Identify and resolve deadlocks by aborting one of the conflicting transactions.

Database Recovery

Introduction:

It refers to the process of restoring a database to a consistent and usable state after a failure or error occurs.

It involves recovering lost or corrupted data and ensuring data integrity and consistency.

Need for Recovery

Data Loss :

Accidental deletion, hardware failures, or software errors can result in data loss.

System Crashes :

Power outages, hardware failures, or software bugs can cause the database system to crash, leading to data corruption.

Human Errors :

Incorrect data entry, unauthorized modifications, or improper operations can compromise data integrity.

Types of Errors

Logical Errors :

Data inconsistencies or incorrect data entries due to application or user errors.

Physical Errors :

Hardware failures, disk crashes, or network failures that result in data loss or corruption.

System Errors:

Software bugs, database crashes, or operating system failures that impact database operations.

Recovery Techniques

1. **Backup and Restore**

Full Backup :

A complete backup of the database.

Incremental Backup :

Backup of changes made since the last backup.

Differential Backup :

Backup of changes made since the last full backup.

Point-in-Time Recovery :

Restoring the database to a specific point in time using backup files.

2. **Transaction Logs**

Write-Ahead Logging (WAL) :

Log records changes before they are applied to the database, ensuring that modifications are recoverable.

Redo Logs :

Contains a record of changes made to the database and can be used to reapply changes after a crash.

Undo Logs :

Contains a record of changes that can be used to roll back transactions or undo changes.

3. **Checkpoints**

Database Checkpoints :

Records the state of the database at a specific point in time and can be used to speed up recovery.

Fuzzy Checkpoints :

Allows transactions to continue while the checkpoint is in progress, reducing downtime.

4. **Rollback and Rollforward**

Rollback:

Reverts transactions that were not committed before a crash, ensuring data consistency.

Rollforward :

Applies changes recorded in transaction logs after a crash to bring the database to a consistent state.

5. **Database Mirroring and Replication**

Mirroring:

Maintains a synchronized copy of the database on a secondary server, ensuring data availability and fault tolerance.

Replication:

Copies data from one database to another, providing redundancy and scalability.

6. **RAID (Redundant Array of Independent Disks)**

RAID Levels :

RAID configurations such as RAID 1 (mirroring) and RAID 5 (striping with parity) provide data redundancy and fault tolerance.

7**. Database Archiving**

Archive and Purge :

Moves inactive or historical data to an archive database, reducing the size of the active database and improving performance.

8. **Disaster Recovery Planning**

Backup Sites :

Establishes backup sites or data centers to restore operations in case of a disaster.

Failover and Redundancy :

Implements failover mechanisms and redundant systems to minimize downtime.