Software Engineering & SDLC (Software Development Life Cycle) Phases

1. Life Cycle Phases: The Software Development Life Cycle (SDLC) comprises several phases, each with its own set of activities and deliverables. These phases ensure a structured and methodical approach to software development. The typical phases include:

- **Planning**: Establishing the scope, objectives, and feasibility of the project.
- **Requirements Analysis**: Gathering and analyzing the functional and non-functional requirements.
- **Design and Prototyping**: Creating the architecture and design models, and developing prototypes.
- **Development**: Writing the actual code based on the design.
- **Testing**: Verifying that the software works as intended and is free of defects.
- **Deployment**: Releasing the software to the users and deploying it in the production environment.
- **Maintenance**: Performing ongoing support, updates, and bug fixes.

2. Planning Analysis: Planning is a critical phase where project goals, timelines, resources, and risks are identified and documented. It involves:

- Defining the project scope and objectives.
- Conducting feasibility studies.
- Developing project plans, including timelines and resource allocations.
- Identifying potential risks and mitigation strategies.

3. Requirements Analysis: Requirements analysis involves understanding the needs and constraints of the stakeholders and documenting them. Key activities include:

- Eliciting requirements through interviews, surveys, and observations.
- Analyzing and prioritizing requirements.

- Creating detailed requirement specifications.
- Validating requirements with stakeholders.

4. Design and Prototyping: During the design phase, the system's architecture and components are defined. Prototyping can be used to refine requirements and design choices. Activities include:

- Creating system architecture and design models.
- Developing prototypes to validate design choices.
- Reviewing and refining the design.

5. Development of the Application: This phase involves the actual coding of the software. Developers translate design documents into code. Activities include:

- Writing code in the chosen programming language.
- Integrating different components of the software.
- Conducting code reviews and unit testing.

6. Testing and Deployment: Testing ensures that the software is of high quality and meets the requirements. Deployment involves releasing the software to users. Activities include:

- Performing various levels of testing (unit, integration, system, acceptance).
- Identifying and fixing defects.
- Preparing deployment plans.
- Deploying the software to the production environment.
- Conducting post-deployment validation.

7. Project Management: Project management involves planning, executing, and closing projects. It ensures that project goals are met on time and within budget. Key activities include:

- Planning project schedules and resources.
- Managing project risks and issues.
- Monitoring project progress and performance.
- Communicating with stakeholders.
- Closing the project and documenting lessons learned.

10 MCQs with Keys:

1. **Which phase of the SDLC involves gathering and analyzing the functional and non-functional requirements?**

   a) Planning

   b) Design

   c) Requirements Analysis

   d) Development

2. **What is the main objective of the planning phase in the SDLC?**

   a) Writing code

   b) Testing the software

   c) Establishing the scope, objectives, and feasibility of the project

   d) Deploying the software

3. **Which phase involves creating the architecture and design models of the software?**

   a) Requirements Analysis

   b) Design and Prototyping

   c) Development

   d) Maintenance

4. **In which phase is the actual code for the software written?**

   a) Planning

   b) Development

   c) Testing

d) Deployment

5. **Which phase focuses on verifying that the software works as intended and is free of defects?**

    a) Development

    b) Testing

    c) Design

    d) Requirements Analysis

6. **During which phase is the software released to the users?**

    a) Planning

    b) Deployment

    c) Maintenance

    d) Requirements Analysis

7. **What is a key activity during the project management phase of the SDLC?**

    a) Writing code

    b) Monitoring project progress and performance

    c) Conducting unit testing

    d) Developing prototypes

8. **What is the purpose of creating prototypes during the design phase?**

    a) To gather requirements

    b) To validate design choices

    c) To deploy the software

d) To monitor project progress

9. **Which phase involves ongoing support, updates, and bug fixes after the software is deployed?**

    a) Development

    b) Testing

    c) Maintenance

    d) Design

10. **Which document typically results from the requirements analysis phase?**

    a) Project plan

    b) Design specification

    c) Requirements specification

    d) Test plan

Pre-code Planning: Flow Chart and Pseudocode

1. Pseudocode: Pseudocode is a simplified, half-code, half-English way of writing algorithms. It allows programmers to outline their ideas before converting them into actual code. It is not bound by syntax rules of any specific programming language, making it easier to understand and communicate.

2. Verify Algorithm: Verifying an algorithm involves ensuring that it is correct and efficient. This can be done through techniques such as:

- **Dry Running:** Manually going through the algorithm step-by-step with sample inputs.

- **Peer Review:** Having other developers review the algorithm for logical errors.
- **Testing:** Implementing the algorithm and testing it with various inputs to ensure it produces the correct outputs.

3. Flowchart: A flowchart is a graphical representation of an algorithm. It uses different symbols to represent different types of actions or steps in a process. Common symbols include:

- **Ovals:** Represent start and end points.
- **Rectangles:** Represent process steps.
- **Diamonds:** Represent decision points.
- **Arrows:** Indicate the flow of the process.

10 MCQs with Keys on Algorithm Topics

1. **What is the primary purpose of pseudocode?**

   a) To compile the code

   b) To outline the logic of an algorithm in a human-readable format

   c) To execute the program

   d) To create graphical representations

2. **Which of the following is NOT typically included in pseudocode?**

   a) Specific syntax of a programming language

   b) Descriptive variable names

   c) Logical steps of an algorithm

   d) Conditional statements

3. **What is the purpose of verifying an algorithm?**

   a) To convert it into code

b) To ensure it is correct and efficient

c) To make it more complex

d) To visualize the process

4. **Which technique involves manually going through an algorithm with sample inputs to check its correctness?**

    a) Debugging

    b) Dry Running

    c) Compiling

    d) Flowcharting

5. **In a flowchart, which symbol is typically used to represent the start and end points?**

    a) Rectangle

    b) Diamond

    c) Oval

    d) Arrow

6. **In a flowchart, which symbol represents a decision point?**

    a) Oval

    b) Rectangle

    c) Diamond

    d) Arrow

7. **What does an arrow represent in a flowchart?**

    a) A process step

b) A start point

c) A decision point

d) The flow of the process

8. **Which of the following is a benefit of using pseudocode?**

a) It can be directly executed by a computer

b) It helps programmers think through the logic before coding

c) It requires knowledge of specific programming syntax

d) It replaces the need for a flowchart

9. **How can peer review help in verifying an algorithm?**

a) By writing the algorithm in code

b) By visually representing the algorithm

c) By having other developers check for logical errors

d) By running the algorithm with test inputs

10. **Which symbol in a flowchart is used to represent a process or action step?**

a) Oval

b) Rectangle

c) Diamond

d) Arrow

Git and Related Topics

1. What is Git? Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency. It allows multiple developers to work on the same project simultaneously without interfering with each other's work.

2. How to Install Git on Windows?

- Download the Git installer from the official Git website.
- Run the installer and follow the setup instructions.
- Choose default settings for most of the installation process.
- Verify the installation by opening Command Prompt and typing git --version.

3. What is GitHub? GitHub is a web-based platform that uses Git for version control. It allows developers to collaborate on projects, share code, and manage repositories online.

4. Git Commands

- git init: Initializes a new Git repository.
- git add: Adds files to the staging area.
- git commit: Commits changes to the repository.
- git status: Shows the status of changes.
- git log: Displays the commit history.
- git branch: Manages branches.
- git merge: Merges branches.
- git clone: Clones a repository.
- git pull: Fetches and integrates changes from a remote repository.
- git push: Pushes local changes to a remote repository.

5. Git vs. GitHub

- Git is a version control system.
- GitHub is a platform that hosts Git repositories and provides tools for collaboration.

6. What is GitLab? GitLab is a web-based DevOps lifecycle tool that provides a Git repository manager providing wiki, issue-tracking, and CI/CD pipeline features, using an open-source license.

7. Git Clone Commands

- git clone <repository_url>: Clones a repository into a new directory.

8. Git Push Commands

- git push origin <branch_name>: Pushes changes to the specified branch on the remote repository.

9. Git Pull Commands

- git pull origin <branch_name>: Fetches and merges changes from the specified branch on the remote repository.

10. Git History - git log: Shows the commit history. - git reflog: Shows the reference logs of all actions performed in the repository.

11. Branching and Merging - git branch <branch_name>: Creates a new branch. - git checkout <branch_name>: Switches to the specified branch. - git merge <branch_name>: Merges the specified branch into the current branch.

12. Resolve Merge Conflicts in Git - When conflicts occur during a merge, Git marks the conflict areas. - Open the affected files and manually resolve the conflicts. - Use git add <file> to mark the conflicts as resolved. - Commit the resolved changes with git commit.

10 MCQs with Keys

1. **What is Git?**

   a) A programming language

   b) A web hosting service

c) A version control system

d) An integrated development environment (IDE)

2. **Which command initializes a new Git repository?**

    a) git start

    b) git init

    c) git create

    d) git new

3. **What does git add do?**

    a) Commits changes to the repository

    b) Adds files to the staging area

    c) Shows the status of changes

    d) Creates a new branch

4. **What is GitHub?**

    a) A version control system

    b) A Git repository manager

    c) A CI/CD pipeline tool

    d) An issue-tracking software

5. **How do you push changes to a remote repository in Git?**

    a) git commit

    b) git add

    c) git push

d) git fetch

6. **Which command is used to clone a repository in Git?**

    a) git copy

    b) git fork

    c) git clone

    d) git duplicate

7. **What does git pull do?**

    a) Fetches changes from a remote repository and merges them

    b) Pushes changes to a remote repository

    c) Adds files to the staging area

    d) Commits changes to the repository

8. **What is the purpose of branching in Git?**

    a) To create a copy of a repository

    b) To manage different versions of a project

    c) To push changes to a remote repository

    d) To resolve conflicts

9. **What does git merge do?**

    a) Combines the changes from different branches

    b) Deletes a branch

    c) Creates a new branch

    d) Initializes a new repository

10. **How are merge conflicts resolved in Git?**

a) By using the git fix command

b) By manually editing the conflicted files

c) By re-cloning the repository

d) By running git resolve

RDBMS

Database Management System (DBMS)

1. What is a DBMS? A Database Management System (DBMS) is software that uses a standard method to store and organize data. It allows users to create, read, update, and delete data in a database.

2. Components of a DBMS:

- **Hardware:** Physical devices.
- **Software:** DBMS software itself, operating system, network software.
- **Data:** The actual data stored in the database.
- **Procedures:** Instructions and rules for using the database.
- **Database Access Language:** SQL for accessing and manipulating data.
- **Users:** Database administrators, developers, end-users.

3. Advantages for Users:

- **Data Abstraction and Independence:** Simplifies data access.
- **Data Security:** Ensures only authorized users can access data.
- **Data Integrity:** Maintains accuracy and consistency of data.
- **Concurrent Access:** Multiple users can access data simultaneously.
- **Backup and Recovery:** Automated backup and recovery processes.

Features and Characteristics of Database Models

1. Flat-file Model:

- **Features:** Simple, stores data in plain text files.
- **Characteristics:** Each line in a flat file holds a single record, and fields are separated by delimiters like commas or tabs.
- **Usage:** Suitable for simple, small-scale applications with limited relationships between data.

2. Hierarchical Model:

- **Features:** Data is organized in a tree-like structure.
- **Characteristics:** Each record has a single parent and can have multiple children. Relationships are defined in a parent-child hierarchy.
- **Usage:** Suitable for applications with one-to-many relationships like organizational structures.

3. XML Model:

- **Features:** Uses XML format to store data.
- **Characteristics:** Flexible, platform-independent, and allows complex data structures. Hierarchical but can represent more complex relationships.
- **Usage:** Suitable for web applications and data interchange between systems.

Levels of a DBMS Architecture

1. **Internal Level:** Physical storage of data.
2. **Conceptual Level:** Logical structure of the entire database.
3. **External Level:** Individual user views of the database.

Types of Constraints

1. **Domain Constraints:** Restrict the type of data.
2. **Entity Integrity Constraints:** Ensure that each table has a primary key and that the key is unique and not null.

3. **Referential Integrity Constraints:** Ensure that foreign keys correctly and consistently refer to primary keys.

Normalization

1. Normalization: Normalization is a database design technique that reduces data redundancy and improves data integrity by organizing fields and table relationships.

2. First Normal Form (1NF):

- **Requirement:** Eliminate duplicate columns, create separate tables for each group of related data, and identify each set of related data with a primary key.
- **Example:** Convert a table with repeating groups into multiple tables, each with a primary key.

3. Second Normal Form (2NF):

- **Requirement:** Must be in 1NF and all non-key attributes must be fully functional dependent on the primary key.
- **Example:** Remove partial dependencies; ensure each non-key attribute is dependent on the entire primary key.

4. Third Normal Form (3NF):

- **Requirement:** Must be in 2NF and all attributes must be directly dependent on the primary key.
- **Example:** Remove transitive dependencies; non-key attributes should not depend on other non-key attributes.

5. Boyce-Codd Normal Form (BCNF):

- **Requirement:** A stronger version of 3NF; every determinant must be a candidate key.
- **Example:** Resolve anomalies by ensuring that for every functional dependency ($X \rightarrow Y$), X is a super key.

10 MCQs with Keys

1. **What is a DBMS?**

    a) A hardware component

    b) Software that manages databases

    c) A programming language

    d) A network protocol

2. **Which component of a DBMS handles physical storage?**

    a) Software

    b) Hardware

    c) Data

    d) Procedures

3. **Which database model uses a tree-like structure?**

    a) Flat-file

    b) Hierarchical

    c) XML

    d) Relational

4. **What is the purpose of normalization in database design?**

    a) To increase redundancy

    b) To simplify data structures

    c) To reduce data redundancy and improve data integrity

    d) To create more tables

5. **Which of the following is a feature of the XML database model?**

a) Simple and limited relationships

b) Tree-like hierarchical structure

c) Platform-independent and allows complex data structures

d) Uses primary and foreign keys

6. **What level of DBMS architecture represents the physical storage of data?**

   a) External level

   b) Conceptual level

   c) Internal level

   d) Logical level

7. **Which type of constraint ensures a field must have unique and non-null values?**

   a) Domain constraint

   b) Entity integrity constraint

   c) Referential integrity constraint

   d) Unique constraint

8. **What does 1NF require for a database design?**

   a) Eliminate transitive dependencies

   b) Create separate tables for groups of related data

   c) Ensure non-key attributes are dependent on the primary key

   d) Ensure all determinants are candidate keys

9. **In which normal form must a database be if it is free of partial dependencies?**

a) 1NF

b) 2NF

c) 3NF

d) BCNF

10. **What is required for a database to be in BCNF?**

a) It must be in 1NF

b) It must be in 2NF

c) It must be in 3NF and every determinant must be a candidate key

d) It must eliminate domain constraints

ERD

Entity-Relationship Modeling for a RDBMS

1. Entity-Relationship Modeling: Entity-Relationship (ER) modeling is a diagrammatic approach to database design. It uses entities, attributes, and relationships to represent data and its interconnections.

2. Definitions:

- **Entities:** Objects or things in the real world that have an independent existence and can be distinctly identified. Examples include Customer, Order, Product.
- **Attributes:** Properties or characteristics of entities. Examples include CustomerID, CustomerName for the Customer entity.
- **Relationships:** Associations among entities. Examples include Customer places Order, Product belongs to Category.

3. Degree of Relationships:

- The degree of a relationship refers to the number of entities involved in the relationship.
- **Unary (Degree 1):** A relationship involving a single entity (e.g., an employee manages other employees).
- **Binary (Degree 2):** A relationship involving two entities (e.g., customers place orders).
- **Ternary (Degree 3):** A relationship involving three entities (e.g., a doctor prescribes medication to a patient).

4. Cardinality of Relationships:

- Cardinality specifies the number of instances of one entity that can or must be associated with each instance of another entity.
- **One-to-One (1:1):** Each entity instance in the relationship will have exactly one related instance (e.g., a person has one passport).
- **One-to-Many (1**

  ): An entity instance on one side of the relationship can be associated with multiple instances on the other side (e.g., a customer places multiple orders).

- **Many-to-Many (M**

  ): Instances on both sides of the relationship can have multiple associations (e.g., students enroll in multiple courses, and courses have multiple students).

5. Relational Database Model:

- A relational database organizes data into tables (relations) that are composed of rows and columns.
- Each table represents an entity and each row represents a single instance of that entity.
- Relationships are represented through foreign keys.

6. Create an ERD (Entity-Relationship Diagram) for a Database Based on a Scenario:

Scenario:

- A university database includes information about students, courses, and instructors.
- Students enroll in courses.
- Instructors teach courses.
- Courses have multiple students and are taught by multiple instructors.

10 MCQs and Keys

1. **What is an entity in an ER model?**

   a) A property of an object

   b) A uniquely identifiable object in the real world

   c) An association among objects

   d) A value in a database table

2. **Which of the following is an attribute?**

   a) Order

   b) CustomerID

   c) Enrolls

   d) Teaches

3. **What is a relationship in an ER model?**

   a) A property of an object

   b) A uniquely identifiable object in the real world

   c) An association among entities

   d) A value in a database table

4. **What does the degree of a relationship indicate?**

a) The type of relationship

b) The number of entities involved in a relationship

c) The cardinality of a relationship

d) The attributes of entities

5. **Which of the following describes a one-to-many relationship?**

a) A student enrolls in one course

b) A course has one instructor

c) A customer places multiple orders

d) A person has one passport

6. **What is the cardinality of a many-to-many relationship?**

a) One-to-One

b) One-to-Many

c) Many-to-One

d) Many-to-Many

7. **In a relational database, how are entities represented?**

a) As rows in tables

b) As columns in tables

c) As tables

d) As databases

8. **In an ERD, what does a diamond shape typically represent?**

a) An entity

b) An attribute

c) A relationship

d) A primary key

9. **Which type of relationship involves three entities?**

a) Unary

b) Binary

c) Ternary

d) Quaternary

10. **In a university database ERD, which relationship best describes the scenario "students enroll in courses"?**

a) One-to-One

b) One-to-Many

c) Many-to-Many

d) Many-to-One

==========

SQL with MySQL

Understanding Databases and SQL

1. What is a Database? A database is an organized collection of structured information, or data, typically stored electronically in a computer system. Databases are managed by Database Management Systems (DBMS).

2. What is SQL? SQL (Structured Query Language) is a standard programming language specifically designed for managing and

manipulating relational databases. It allows users to create, read, update, and delete database records.

3. What is MySQL? MySQL is an open-source relational database management system (RDBMS) that uses SQL. It is one of the most popular databases for web applications.

SQL Commands

1. Types of SQL Commands:

- **Data Definition Language (DDL):** Commands that define the structure of the database (e.g., CREATE, ALTER, DROP).
- **Data Manipulation Language (DML):** Commands that manipulate the data stored in the database (e.g., INSERT, UPDATE, DELETE).
- **Data Query Language (DQL):** Command that queries the database (e.g., SELECT).
- **Data Control Language (DCL):** Commands that control access to the data (e.g., GRANT, REVOKE).
- **Transaction Control Language (TCL):** Commands that deal with transaction management (e.g., COMMIT, ROLLBACK).

Database Connection

1. Launching MySQL Workbench: MySQL Workbench is a visual tool for database design and management. To launch it:

- Open MySQL Workbench from your applications or start menu.

2. Connecting to MySQL Server:

- Open MySQL Workbench.
- Click on the "+" symbol to create a new connection.
- Enter the connection details (hostname, port, username, password).
- Click "Test Connection" to verify the connection.
- Save and click on the newly created connection to connect to the server.

3. Creating a New Database:

- In MySQL Workbench, connect to your MySQL server.
- Click on the "SQL Editor" tab.
- Execute the SQL command: CREATE DATABASE database_name;

4. Data Types:

- **Numeric:** INT, FLOAT, DOUBLE, DECIMAL
- **String:** CHAR, VARCHAR, TEXT
- **Date and Time:** DATE, TIME, DATETIME, TIMESTAMP
- **Others:** BOOLEAN, ENUM, BLOB

5. CAST or CONVERT:

- CAST(expression AS data_type): Converts an expression from one data type to another.
- CONVERT(expression, data_type): Another way to convert data types (specific to MySQL).

6. Keys in SQL:

- **Primary Key:** Uniquely identifies each record in a table.
- **Foreign Key:** A field (or collection of fields) in one table that refers to the primary key in another table.
- **Unique Key:** Ensures all values in a column are unique.

7. Constraints:

- **NOT NULL:** Ensures a column cannot have a NULL value.
- **UNIQUE:** Ensures all values in a column are unique.
- **PRIMARY KEY:** Uniquely identifies each row in a table.
- **FOREIGN KEY:** Ensures referential integrity for a record in another table.
- **CHECK:** Ensures all values in a column satisfy a specific condition.
- **DEFAULT:** Sets a default value for a column if no value is specified.

10 MCQs with Keys

1. **What is a database?**

a) A programming language

b) A collection of interrelated data

c) A spreadsheet

d) An operating system

2. **What does SQL stand for?**

    a) Structured Query Language

    b) Standard Query Language

    c) Simple Query Language

    d) System Query Language

3. **Which of the following is an open-source relational database management system?**

    a) SQL Server

    b) MySQL

    c) Oracle

    d) IBM DB2

4. **Which SQL command is used to retrieve data from a database?**

    a) SELECT

    b) INSERT

    c) UPDATE

    d) DELETE

5. **How do you create a new database in MySQL?**

    a) NEW DATABASE database_name;

b) CREATE DB database_name;

c) CREATE DATABASE database_name;

d) ADD DATABASE database_name;

6. **Which data type is used to store large text data in MySQL?**

   a) VARCHAR

   b) CHAR

   c) TEXT

   d) STRING

7. **What does the CAST function do in SQL?**

   a) Deletes a column

   b) Converts data from one type to another

   c) Creates a new table

   d) Adds a new row

8. **What is the purpose of a primary key in a database table?**

   a) To allow duplicate values

   b) To uniquely identify each record

   c) To store text data

   d) To join two tables

9. **Which SQL constraint ensures that all values in a column are unique?**

   a) NOT NULL

   b) DEFAULT

c) UNIQUE

d) CHECK

10. **What is the role of a foreign key in a database?**

a) To uniquely identify each record

b) To enforce referential integrity

c) To store dates and times

d) To allow NULL values

==========

DDL Commands and Query Clauses in SQL

1. DDL Commands: Data Definition Language (DDL) commands are used to define and modify database objects such as tables.

2. Add Table to Database:

- CREATE TABLE table_name (column1 datatype, column2 datatype, ...);

3. Describe Table:

- DESCRIBE table_name; or SHOW COLUMNS FROM table_name;

4. Alter Table:

- **Add a Column:** ALTER TABLE table_name ADD column_name datatype;
- **Modify a Column:** ALTER TABLE table_name MODIFY column_name new_datatype;
- **Drop a Column:** ALTER TABLE table_name DROP COLUMN column_name;

5. Modify and Drop Clause:

- **Modify:** ALTER TABLE table_name MODIFY column_name datatype;
- **Drop Table:** DROP TABLE table_name;

6. Data Manipulation:

- **Insert Data:** INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);
- **Update Data:** UPDATE table_name SET column1 = value1, column2 = value2 WHERE condition;
- **Delete Data:** DELETE FROM table_name WHERE condition;
- **Select Data:** SELECT column1, column2, ... FROM table_name WHERE condition;

Query Clauses and Other Concepts

1. Database Schema:

- The structure that represents the logical view of the entire database.

2. Import Data:

- Using SQL commands or tools to load data into a database.

3. Query Clauses:

- **WHERE:** Filters records based on specified conditions.
- **ORDER BY:** Sorts the result set in ascending or descending order.
- **GROUP BY:** Groups rows that have the same values in specified columns into summary rows.
- **HAVING:** Filters records that work on summarized GROUP BY results.
- **LIMIT:** Specifies the number of records to return.

4. Column Alias and Table Alias:

- **Column Alias:** SELECT column_name AS alias_name FROM table_name;
- **Table Alias:** SELECT t.column_name FROM table_name AS t;

10 MCQs with Keys

1. **Which command is used to create a new table in a database?**

   a) CREATE DATABASE

   b) CREATE TABLE

   c) ADD TABLE

   d) INSERT TABLE

2. **How do you view the structure of a table?**

   a) SHOW TABLE

   b) VIEW TABLE

   c) DESCRIBE table_name

   d) LIST TABLE

3. **Which command adds a new column to an existing table?**

   a) ALTER TABLE table_name ADD column_name datatype

   b) MODIFY TABLE table_name ADD column_name datatype

   c) CREATE COLUMN column_name datatype

   d) ADD COLUMN column_name datatype

4. **How do you change the data type of an existing column in a table?**

   a) ALTER TABLE table_name CHANGE column_name new_datatype

b) ALTER TABLE table_name MODIFY column_name new_datatype

c) ALTER TABLE table_name ALTER column_name new_datatype

d) MODIFY TABLE table_name CHANGE column_name new_datatype

5. **Which command removes a table from the database?**

   a) REMOVE TABLE table_name

   b) DELETE TABLE table_name

   c) DROP TABLE table_name

   d) ERASE TABLE table_name

6. **What SQL command is used to insert data into a table?**

   a) INSERT INTO table_name (columns) VALUES (values)

   b) ADD DATA table_name (columns) VALUES (values)

   c) INSERT DATA table_name (columns) VALUES (values)

   d) ADD INTO table_name (columns) VALUES (values)

7. **Which clause is used to filter records in a SELECT statement?**

   a) FILTER BY

   b) WHERE

   c) GROUP BY

   d) HAVING

8. **How do you sort the result set in descending order?**

   a) SORT DESC

b) ORDER BY column_name DESC

c) SORT BY column_name DESC

d) ORDER column_name DESC

9. **Which clause groups rows that have the same values into summary rows?**

a) WHERE

b) GROUP BY

c) HAVING

d) ORDER BY

10. **How do you assign an alias to a column in a SELECT statement?**

a) SELECT column_name alias_name FROM table_name

b) SELECT column_name TO alias_name FROM table_name

c) SELECT column_name AS alias_name FROM table_name

d) SELECT column_name LIKE alias_name FROM table_name

Understanding Joins in SQL

1. Introduction to Joins: Joins are used in SQL to combine rows from two or more tables based on a related column between them.

2. Types of Joins:

- **Inner Join**
- **Left Outer Join**

- **Right Outer Join**
- **Full Outer Join**
- **Self-Join**
- **Equi Join**
- **Non-equi Join**

3. Inner Join:

- Retrieves records that have matching values in both tables.
- Syntax: SELECT columns FROM table1 INNER JOIN table2 ON table1.column = table2.column;

4. Left Outer Join (Left Join):

- Retrieves all records from the left table and the matched records from the right table. Unmatched records will have NULL in the columns from the right table.
- Syntax: SELECT columns FROM table1 LEFT JOIN table2 ON table1.column = table2.column;

5. Right Outer Join (Right Join):

- Retrieves all records from the right table and the matched records from the left table. Unmatched records will have NULL in the columns from the left table.
- Syntax: SELECT columns FROM table1 RIGHT JOIN table2 ON table1.column = table2.column;

6. Full Outer Join:

- Retrieves all records when there is a match in either left or right table. Records without a match in either table will have NULLs for the columns from the table without the match.
- Syntax: SELECT columns FROM table1 FULL OUTER JOIN table2 ON table1.column = table2.column;

7. ANSI Join Syntax:

- Standard syntax used for joins across different RDBMS.

- Example: SELECT columns FROM table1 INNER JOIN table2 ON table1.column = table2.column;

8. Self-Join:

- A self-join is a regular join but the table is joined with itself.
- Syntax: SELECT a.column1, b.column2 FROM table a, table b WHERE a.common_column = b.common_column;

9. Equi and Non-equi Join:

- **Equi Join:** Uses equality operator (=) to join tables.
- **Non-equi Join:** Uses other operators like !=, <, >, etc., to join tables.

10. Set Operations:

- **UNION:** Combines the result set of two or more SELECT statements (removes duplicates).
- **UNION ALL:** Combines the result set of two or more SELECT statements (includes duplicates).
- **INTERSECT:** Returns only the records that are common to both SELECT statements.
- **EXCEPT (MINUS):** Returns the records from the first SELECT statement that are not in the second.

10 MCQs with Keys

1. **What does an inner join return?**

   a) All records from the left table

   b) All records from the right table

   c) Only the matched records from both tables

   d) All records from both tables

Which SQL statement represents a left outer join?

a) SELECT * FROM table1 INNER JOIN table2 ON table1.id = table2.id;

b) SELECT * FROM table1 LEFT JOIN table2 ON table1.id = table2.id;

c) SELECT * FROM table1 RIGHT JOIN table2 ON table1.id = table2.id;

d) SELECT * FROM table1 FULL OUTER JOIN table2 ON table1.id = table2.id;

2. **What does a right outer join return?**

   a) Only the matched records from both tables

   b) All records from the left table

   c) All records from the right table

   d) All records from the right table and matched records from the left table

3. **Which join would you use to get all records that are in both tables or in either table?**

   a) Inner Join

   b) Left Join

   c) Right Join

   d) Full Outer Join

4. **What is a self-join?**

   a) A join of a table with another table

   b) A join of a table with itself

   c) A join that returns only distinct rows

d) A join that combines columns from two different databases

5. **What is an equi join?**

   a) A join using any comparison operator except =

   b) A join that uses the equality operator (=) for matching rows

   c) A join that returns unmatched rows from both tables

   d) A join that combines more than two tables

6. **Which set operation combines the result sets of two queries and removes duplicates?**

   a) UNION

   b) UNION ALL

   c) INTERSECT

   d) EXCEPT

7. **Which SQL clause can be used to give a table an alias?**

   a) AS

   b) LIKE

   c) WITH

   d) BY

8. **How do you combine rows from two tables based on a related column between them?**

   a) SELECT

   b) JOIN

   c) WHERE

d) GROUP BY

9. **What type of join uses operators other than = to join tables?**

   a) Equi Join

   b) Non-equi Join

   c) Inner Join

   d) Self-Join

10. What type of join uses operators other than = to join tables?

a) Equi Join

b) Non-equi Join

c) Inner Join

d) Self-Join

SQL Functions and Subqueries

1. String Functions:

- Functions that operate on string data types.
- Examples:
  - CONCAT(str1, str2, ...): Concatenates two or more strings.
  - SUBSTRING(str, pos, len): Extracts a substring from a string.
  - UPPER(str): Converts a string to uppercase.
  - LOWER(str): Converts a string to lowercase.
  - LENGTH(str): Returns the length of a string.

2. Numeric Functions:

- Functions that perform operations on numeric data types.
- Examples:

    - ABS(number): Returns the absolute value of a number.
    - CEIL(number): Rounds a number up to the nearest integer.
    - FLOOR(number): Rounds a number down to the nearest integer.
    - ROUND(number, decimals): Rounds a number to a specified number of decimal places.
    - MOD(number, divisor): Returns the remainder of a division.

3. Date Functions:

- Functions that operate on date data types.
- Examples:

    - NOW(): Returns the current date and time.
    - CURDATE(): Returns the current date.
    - DATE_ADD(date, INTERVAL value unit): Adds a time interval to a date.
    - DATEDIFF(date1, date2): Returns the difference in days between two dates.
    - EXTRACT(unit FROM date): Extracts a part of a date.

4. Aggregate Functions:

- Functions that operate on a set of values and return a single value.
- Examples:

    - SUM(column): Returns the sum of a column.
    - AVG(column): Returns the average value of a column.
    - COUNT(column): Returns the number of values in a column.
    - MAX(column): Returns the maximum value in a column.
    - MIN(column): Returns the minimum value in a column.

5. Generate Groups:

- Grouping data based on one or more columns using the GROUP BY clause.

- Example: SELECT column1, SUM(column2) FROM table GROUP BY column1;

SQL Subqueries

1. SQL Subqueries:

- A subquery is a query nested inside another query.
- Example: SELECT * FROM table WHERE column = (SELECT column FROM table2 WHERE condition);

2. Correlated Subqueries:

- A subquery that refers to columns in the outer query.
- Example: SELECT a.column1 FROM table1 a WHERE a.column2 = (SELECT MAX(b.column2) FROM table2 b WHERE b.column1 = a.column1);

3. Non-correlated Subqueries:

- A subquery that does not refer to columns in the outer query and can be executed independently.
- Example: SELECT * FROM table WHERE column IN (SELECT column FROM table2);

10 MCQs with Keys

1. **Which function concatenates two or more strings in SQL?**

   a) CONCAT

   b) SUBSTRING

   c) UPPER

   d) LENGTH

2. **What does the ABS function return?**

   a) The ceiling value of a number

b) The floor value of a number

c) The absolute value of a number

d) The rounded value of a number

3. **Which function returns the current date in SQL?**

    a) NOW()

    b) CURDATE()

    c) DATE_ADD()

    d) DATEDIFF()

4. **What does the SUM function do?**

    a) Returns the maximum value in a column

    b) Returns the sum of a column

    c) Returns the average value of a column

    d) Returns the minimum value in a column

5. **Which clause is used to group rows that have the same values in SQL?**

    a) ORDER BY

    b) GROUP BY

    c) HAVING

    d) WHERE

6. **Which function extracts a part of a date in SQL?**

    a) NOW()

    b) CURDATE()

c) DATE_ADD()

d) EXTRACT

7. **What is a correlated subquery?**

   a) A subquery that can be executed independently of the outer query

   b) A subquery that refers to columns in the outer query

   c) A subquery that always returns a single value

   d) A subquery that uses aggregate functions

8. **Which aggregate function returns the number of values in a column?**

   a) SUM

   b) AVG

   c) COUNT

   d) MAX

9. **What type of subquery can be executed independently of the outer query?**

   a) Correlated subquery

   b) Non-correlated subquery

   c) Aggregate subquery

   d) Scalar subquery

10. **Which function rounds a number to a specified number of decimal places in SQL?**

    a) CEIL

b) FLOOR

c) ROUND

d) MOD

Views, Indexes, Transaction Control Commands, Stored Procedures, and Functions in SQL

1. Views:

- A view is a virtual table based on the result-set of an SQL statement.
- Example: CREATE VIEW view_name AS SELECT column1, column2 FROM table WHERE condition;

2. Index:

- An index is used to speed up the retrieval of rows by using a pointer.
- Example: CREATE INDEX index_name ON table (column1, column2);

3. Transaction Control Commands:

- Commands that manage changes made by DML statements and ensure the integrity of the data.
- Examples:
    - BEGIN TRANSACTION: Starts a transaction.
    - COMMIT: Saves the transaction.
    - ROLLBACK: Undoes the transaction.

4. Stored Procedures:

- A stored procedure is a prepared SQL code that you can save and reuse.
- Example:

```
CREATE PROCEDURE procedure_name
AS
BEGIN
  SQL statements;
END;
```

5. Difference between Procedure and Function:

- **Procedure:**
  - Can perform actions (like INSERT, UPDATE).
  - Does not need to return a value.
  - Example: CREATE PROCEDURE ...
- **Function:**
  - Must return a value.
  - Primarily used for computations.
  - Example: CREATE FUNCTION ... RETURNS ...

6. Creating a Procedure:

- Syntax:

```
CREATE PROCEDURE procedure_name
AS
BEGIN
  SQL statements;
END;
```

7. Creating a Function:

- Syntax:

```
CREATE FUNCTION function_name
RETURNS return_datatype
AS
BEGIN
  SQL statements;
  RETURN return_value;
```

```
END;
```

10 MCQs with Keys

1. **What is a view in SQL?**

   a) A stored procedure

   b) A virtual table based on the result-set of an SQL statement

   c) An index

   d) A physical table

2. **Which command is used to create an index on a table?**

   a) CREATE TABLE

   b) CREATE VIEW

   c) CREATE INDEX

   d) CREATE PROCEDURE

3. **What does the COMMIT command do in SQL?**

   a) Starts a transaction

   b) Saves the transaction

   c) Undoes the transaction

   d) Ends a procedure

4. **Which of the following is used to undo a transaction?**

   a) BEGIN TRANSACTION

   b) COMMIT

   c) ROLLBACK

d) SAVEPOINT

5. **Which statement is true about stored procedures?**

   a) They must return a value

   b) They can perform actions like INSERT and UPDATE

   c) They are only used for calculations

   d) They cannot include SQL statements

6. **What is the main difference between a procedure and a function in SQL?**

   a) A function can perform actions like INSERT and UPDATE

   b) A procedure must return a value

   c) A function must return a value

   d) A procedure is used for computations only

7. **Which SQL statement creates a stored procedure?**

   a) CREATE FUNCTION procedure_name AS BEGIN ... END;

   b) CREATE VIEW procedure_name AS BEGIN ... END;

   c) CREATE PROCEDURE procedure_name AS BEGIN ... END;

   d) CREATE INDEX procedure_name ON ...;

8. **Which SQL statement creates a function?**

   a) CREATE PROCEDURE function_name RETURNS return_datatype AS BEGIN ... END;

   b) CREATE FUNCTION function_name RETURNS return_datatype AS BEGIN ... END;

   c) CREATE VIEW function_name AS BEGIN ... END;

d) CREATE INDEX function_name ON ...;

9. **What is the purpose of an index in SQL?**

a) To create a virtual table

b) To speed up the retrieval of rows by using a pointer

c) To store SQL procedures

d) To manage transactions

10. **Which command is used to start a transaction in SQL?**

a) BEGIN TRANSACTION

b) START TRANSACTION

c) INITIATE TRANSACTION

d) BEGIN TRANSACT