2.1 Blueprinting Database Structures

LEARN

1. Creating Databases and Tables

A database is an organized collection of structured data, typically stored electronically. It enables efficient storage, retrieval, and manipulation of data. Tables within databases organize data into rows and columns, facilitating easy access, management, and querying.

Steps to Create a Database:

Use SQL commands to create a database structure:

```
CREATE DATABASE SchoolDB;
```

This command creates an empty database named SchoolDB.

Creating Tables:

Tables store data in structured formats. Each table consists of columns (attributes) and rows (records).

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    StudentName VARCHAR(50) NOT NULL,
    Age INT CHECK(Age > 0),
    EnrollmentDate DATE DEFAULT CURRENT_DATE
);
```

This creates a table Students with four fields: a unique ID, student name, age (validated to be positive), and enrollment date (defaults to current date).

---

2. Designing Tables with Appropriate Data Types

Selecting the correct data types for columns is crucial for data integrity, efficiency, and optimal performance.

Importance of Correct Data Types:

- **Data Integrity:** Prevents incorrect data entry.
- **Efficient Storage:** Saves storage space and memory usage.
- **Optimized Query Performance:** Enables faster searches and data retrieval.

---

3. Data Types in SQL

SQL offers multiple data types, each suited for different kinds of data:

Data Type	Description	Storage Size	Example
INT	Integer numeric values (without decimals)	4 bytes	Student IDs, Quantity
VARCHAR	Variable-length string	1 byte per character	Names, Addresses
CHAR	Fixed-length string	Fixed size (1 byte per character)	Country codes ('US', 'IN')
DATE	Date values (YYYY-MM-DD)	3 bytes	Birthdates, Enrollment Dates

Data Type	Description	Storage Size	Example
DECIMAL	Numeric data with fixed decimals	Varies	Salaries, Prices
BOOLEAN	True or False	1 byte	Active Status, Flags
TEXT	Large strings of text	Up to 65,535 bytes	Descriptions, Comments

Example of Choosing the Right Data Type:

```
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY,
    EmployeeName VARCHAR(100) NOT NULL,
    Salary DECIMAL(10, 2) CHECK(Salary > 0),
    HireDate DATE,
    IsActive BOOLEAN DEFAULT TRUE
);
```

This design ensures efficiency and correctness by choosing the right data types.

PRACTISE

Task 1: Database and Table Creation

Create a database called BookStore with two tables Books and Authors, specifying relevant data types.

```
CREATE DATABASE BookStore;


CREATE TABLE Authors (
    AuthorID INT PRIMARY KEY,
    AuthorName VARCHAR(100) NOT NULL
);


CREATE TABLE Books (
    BookID INT PRIMARY KEY,
    Title VARCHAR(200) NOT NULL,
    Price DECIMAL(7, 2) CHECK(Price > 0),
    PublicationDate DATE,
    AuthorID INT,
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID)
);
```

Task 2: Choosing Data Types

Create a table named Movies and carefully assign data types based on the type of data it stores:

```
CREATE TABLE Movies (
    MovieID INT PRIMARY KEY,
    Title VARCHAR(200) NOT NULL,
    ReleaseYear INT CHECK(ReleaseYear > 1900),
    Rating DECIMAL(3, 1) CHECK(Rating BETWEEN 0 AND 10),
    Duration INT CHECK(Duration > 0) -- duration in minutes
```

```
);
```

FAQ

- **Q:** Why is choosing the right data type important?
    - **A:** Proper data types ensure accuracy, storage efficiency, and improve database performance.
- **Q:** Can I alter data types after the table creation?
    - **A:** Yes, data types can be modified, but it may risk data loss or require additional adjustments.
- **Q:** Difference between VARCHAR and CHAR?
    - **A:** VARCHAR allocates storage dynamically, while CHAR always uses fixed storage space.
- **Q:** Why is VARCHAR preferred for storing phone numbers?
    - **A:** Because phone numbers often have formatting characters and leading zeros, VARCHAR preserves these accurately.
- **Q:** Are DATE types necessary for dates?
    - **A:** Yes, DATE types are ideal for leveraging database date operations and storage optimization.