Introduction to Relational Databases

Understanding Basic Concepts, Entities, and the Benefits of Relational Databases

Learning Objectives

- Define databases and their key components.
- Understand relational database entities, attributes, and relationships.
- Explore the benefits of relational databases such as data integrity, reduced redundancy, and efficiency.

What is a Database?

- **Definition**: A structured collection of data that can be easily accessed, managed, and updated.
- **Purpose**: Used to store information, manage large datasets, and allow easy access to data.

Types of Databases

- **Relational Databases**: Store data in tables (e.g., SQL, MySQL, PostgreSQL).
- **Non-relational Databases (NoSQL)**: Store data in formats like key-value pairs, document-based (e.g., MongoDB).

Importance of Databases

- **Organization**: Databases organize large amounts of information.
- **Efficiency**: Provide fast data access and retrieval.
- **Consistency**: Ensure consistent data for decision-making and operations.

Introduction to Relational Databases

- **Definition**: A type of database that stores data in **related tables** using predefined relationships.
- **Example**: Tables like "Customers" and "Orders" are related by a common field (CustomerID).

Components of a Relational Database

- **Tables (Entities)**: Store data in rows and columns.
- **Attributes**: Columns that represent characteristics of an entity (e.g., Name, Address).
- **Relationships**: Define how data in one table relates to data in another.

Relational Database Example

- **Entities**: Customers, Orders, Products.
- **Attributes**: CustomerName, OrderID, ProductName.
- **Relationship**: A Customer places multiple Orders.

Entities

- **Definition**: An entity represents an object or thing in the real world that has properties (attributes).
- **Example**: A Customer entity with attributes such as Name, Address, Email.

Attributes

- **Definition**: Attributes are properties that define an entity.
- **Example**: For a Customer, attributes could be CustomerName, Email, Phone.

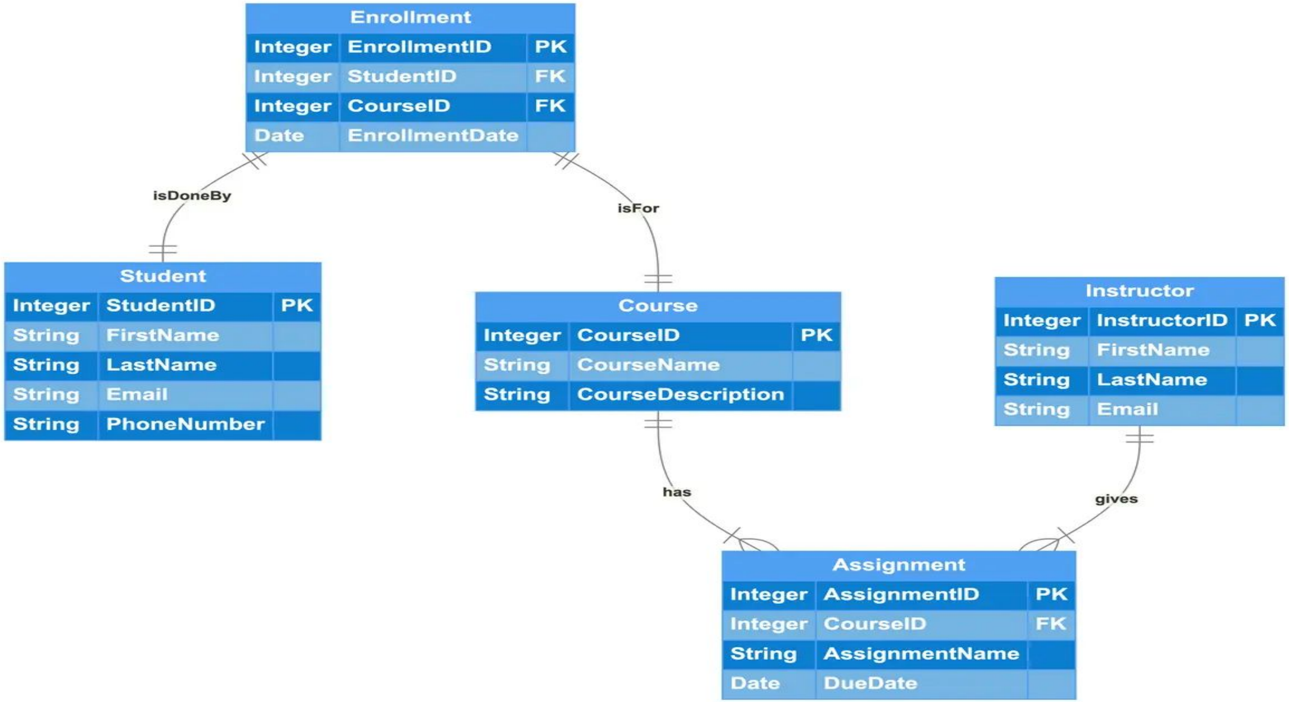
Entity-Relationship Diagrams (ERD)

- **Definition**: A graphical representation of entities, attributes, and relationships.
- **Purpose**: ERDs help visualize the structure of a database.

Example ERD Diagram

Entities: Students, Courses, Instructors.

Relationships: A Student enrolls in Courses, and an Instructor teaches Courses.



Enrollment		
Integer	EnrollmentID	PK
Integer	StudentID	FK
Integer	CourseID	FK
Date	EnrollmentDate	

Student		
Integer	StudentID	PK
String	FirstName	
String	LastName	
String	Email	
String	PhoneNumber	

Course		
Integer	CourseID	PK
String	CourseName	
String	CourseDescription	

Instructor		
Integer	InstructorID	PK
String	FirstName	
String	LastName	
String	Email	

Assignment		
Integer	AssignmentID	PK
Integer	CourseID	FK
String	AssignmentName	
Date	DueDate	

Relationships Between Entities

Definition: Relationships define how entities are connected.

Types:

- One-to-One
- One-to-Many
- Many-to-Many

Types of Relationships in Databases

One-to-One: A person has one passport.

One-to-Many: A Customer places multiple Orders.

Many-to-Many: Students enroll in many Courses, and Courses have many Students.

One-to-One Relationship Example

Example: A **person** entity is related to one **passport** entity.

Use Case: Limited use, but helps avoid redundancy in very specific cases.

One-to-Many Relationship Example

Example: A **customer** can place multiple **orders**.

Use Case: Most common relationship in databases, allows multiple records to be linked to one.

Many-to-Many Relationship Example

Example: A **student** can enroll in many **courses**, and a course can have many students.

Use Case: Requires a **join table** to handle the relationship (e.g., Enrollment table).

Relational Database Management Systems (RDBMS)

Definition: A software that helps manage relational databases.

Examples: MySQL, PostgreSQL, SQL Server.

Key Features of RDBMS

Data Integrity: Ensures the accuracy and consistency of data.

Data Security: Provides user access control and data encryption.

Backup and Recovery: Offers methods to backup and restore data.

Importance of Relational Databases

Flexibility: Relational databases can easily adapt to changes in the system.

Scalability: Support large datasets and handle a growing amount of data efficiently.

Benefits of Relational Databases

Data Integrity: Ensures data is accurate and consistent across tables.

Efficiency: Queries can retrieve large amounts of data quickly using indexes.

Reduced Redundancy: Data is stored in related tables, reducing duplication.

Data Integrity

Definition: The accuracy, consistency, and reliability of data.

Enforced by: Constraints, validation rules, and referential integrity.

How RDBMS Ensures Data Integrity

Constraints: Enforces rules at the database level (e.g., NOT NULL, UNIQUE).

Primary and Foreign Keys: Ensure that relationships between tables remain consistent.

Reduced Redundancy

Explanation: Data is organized into tables based on entities, so no duplicate data is stored.

Example: Instead of storing customer information in every order, store it once in a **Customer** table.

Query Efficiency

SQL Queries: Allow fast data retrieval through optimized searches.

Indexes: Speed up data access by indexing key columns.

Example SQL Query

```
SELECT CustomerName, OrderID

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID
```

Explanation: This query retrieves a list of customer names and their corresponding orders.

Security in Relational Databases

Access Control: Grant or restrict access to different users.

Encryption: Protect sensitive data stored in the database.

Use Cases of Relational Databases

E-commerce: Storing customer and product information.

Healthcare: Managing patient records.

Finance: Tracking transactions and accounts.

Real-World Example: E-commerce

Entities: Products, Customers, Orders.

Relationships: A Customer can place multiple Orders, and each Order can contain multiple Products.