Relational Database

A relational database is a type of database management system (DBMS) that organizes and stores data in a structured manner, with a focus on the relationships between different pieces of data.

It is based on the principles of the relational model, which was developed by Edgar F. Codd in 1970. Relational databases are based on the relational model, an intuitive, straightforward way of representing data in tables.

Relational Database Management System

A Relational Database Management System (RDBMS) is a specific type of database management system that is based on the principles of the relational model. It is a collection of programs designed to create, manage, and maintain relational databases. RDBMS systems are widely used in a variety of applications and industries to store, organize, and retrieve structured data efficiently.

Entity and Attribute

An entity is a piece of data tracked and stored by the system. This could be something as simple as a customer's name and address or more complex information such as an order or invoice.

An entity is typically composed of multiple attributes, the individual data that make up the Entity. Each entity contains some property about its behavior which is also called the attribute.

Example

If we consider a car entity, it can have its attributes such as a car's registration number, car's model, car's name, car's color, number of seats that are there inside the car, etc. Below is the tabular representation of the car entities.

	Registration Number	Model	Name	Colour	Number of Seats
Entity 1 →	DL123	LDI	Ritz	White	5
Entity 2 →	DL234	VDI	Swift	Black	5
Entity 3 →	DL345	ZXI	Dzire	Red	5

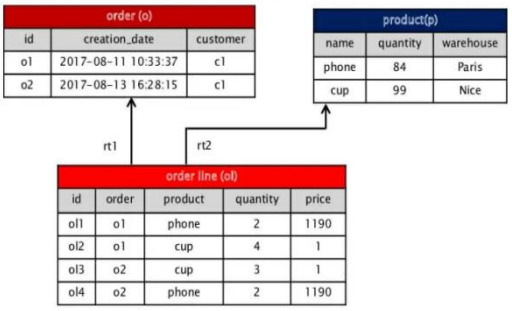
Database Elements

Database elements refer to the fundamental components and structures that make up a database system. These elements work together to store, manage, and retrieve data efficiently. The essential elements of a database system typically include:

- Table
- Fields or Columns
- Records or Rows
- Keys
- Relationships
- Data types

1. Table

A table is the primary unit of physical storage for data in a database. When a user accesses the database, a table is usually referenced for the desired data. Multiple tables might comprise a database, therefore a relationship might exist between tables. Because tables store data, a table requires physical storage on the host computer for the database.

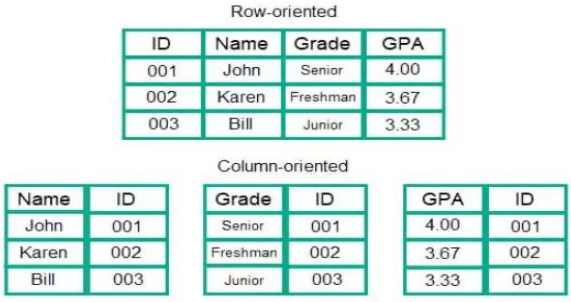


2. Fields or Columns

A column, commonly called a field, is a particular type of data on a table. It is believed that a column only exists in a table, while an attribute exists in an entity.

On the contrary, attributes can be columns, and entities can form a table against the previous immediate claim when a business model transforms into a database model.

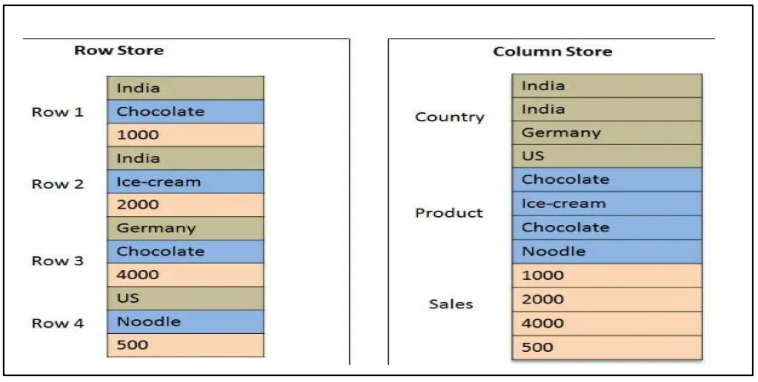
A column is the smallest storage element in any database and depicts related records of a table. A unique data type is given to every column. The data entered into the table will be dependent on the table's data types.



3. Records or Rows

A row of data is the collection of all the columns in a table associated with a single occurrence.

Simply speaking, a row of data is a single record in a table. The population of rows in a table is not constant. A collection of data in a row is also known as a tuple.



4. Keys

A Key is a column or a set of columns in a table that is used to uniquely identify each record and establish relationships between different tables. It ensures the integrity, accuracy, and consistency of the data stored in a database by preventing duplicate or null entries. Keys play a crucial role in maintaining reliable data organization, enforcing constraints, and enabling efficient data retrieval and management within a relational database.

Types of Keys

There are several types of keys used in a database, each serving a specific purpose. Here are the key types commonly used in DBMS:

1. Candidate Key
2. Primary Key
3. Foreign Key
4. Alternate Key
5. Super Key
6. Composite Key

Candidate Key

A candidate key is an attribute or set of attributes that can uniquely identify a row. It must be unique but can contain null values. It ensures that no two rows have the same value for this key and that the key contains no unnecessary attributes.In the given table the attributes Id, SSN, and License No. can be considered as candidate keys.

Id	Name	Address	License No.	SSN
S10000786	Nadim	Khulna	NPR 098765	123-5-908
S10000787	Rana	Dhaka	SDR 043765	983-5-978
S10000788	Maruf	Barisal	NVR 012876	922-5-560
S10000789	Ema	Khulna	QWR090725	654-5-120

Primary Key

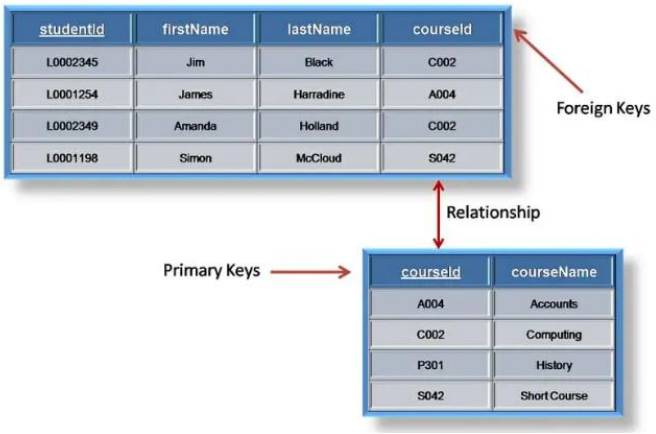
A primary key is the combination of one or more column values in a table that can uniquely identify each row within the table.It must be unique and can not contain null values. Each table can have only one primary key, which may consist of a single column or a combination of multiple columns (composite key). It also acts as a reference point for creating relationships with other tables.

studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042

Foreign Key

A Foreign Key in DBMS is an attribute or a combination of attributes in a table that references the Primary Key of another table, thereby establishing a relationship between the two tables. It is defined in the child

table and ensures that a corresponding parent record exists before a child record can be created, maintaining referential integrity in the database. Similarly, it prevents the deletion of a parent record while related child records still exist. In essence, a foreign key maintains the logical link between related tables and ensures data consistency across the database.



Alternate Key

From the candidate key, one key is selected as the primary key then the rest of the candidate key is called the alternate key. Although it has the same ability to uniquely identify each record, it remains as an alternative option to the primary key. A table can have multiple alternate keys, and they help ensure data uniqueness and integrity within the database. In simple terms, alternate keys act as backup unique identifiers that can be used if needed to access or distinguish records in a table.

In the given table id, License No, and SSN are candidate keys. If Id is selected as primary key then the rest of the key is called the alternate key.

Id	Name	Address	License No.	SSN
S10000686	Alex	Khulna	NPR 098765	123-5-908
S10000687	Akhi	Dhaka	SDR 043765	983-5-978
S10000688	Joti	Barisal	NVR 012876	922-5-560
S10000689	Afrin	Khulna	QWR090725	654-5-120

Super Key

A Super Key in a Database Management System (DBMS) is a set of one or more attributes that can uniquely identify each record in a table. It ensures that no two rows in a relation have the same combination of values for those attributes, thereby maintaining data uniqueness and integrity.

While a super key may contain additional or redundant attributes beyond what is necessary for unique identification, its primary function is to guarantee that every tuple can be distinctly recognized. Every table must have at least one super key, and among all possible super keys, the minimal ones (without unnecessary attributes) are called Candidate Keys, from which a Primary Key is selected.

In the given table, (Id, Name) can be called super key because the names of two employees can be the same, but their ID can't be the same.

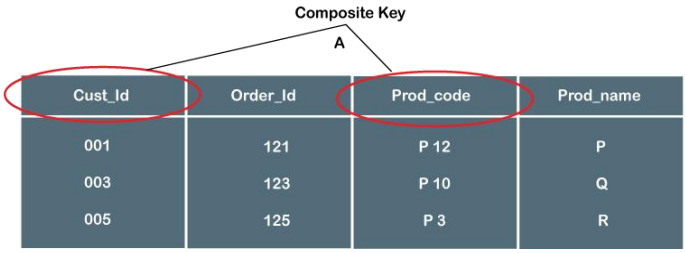
Composite Key

A Composite Key consists of two or more attributes combined to uniquely identify a record in a table,

especially when a single attribute alone cannot ensure uniqueness. Whenever a primary key is formed using multiple attributes, it is known as a Composite Key, also referred to as a Concatenated Key.

It ensures that each record in a relation can be distinctly recognized based on the combined values of these attributes. Composite keys are often used in many-to-many relationship or junction tables, where the uniqueness of a record depends on multiple factors.

For example, in a table named COURSE_REGISTRATION with attributes (Student_ID, Course_ID, Semester), neither Student_ID nor Course_ID alone can uniquely identify a record, but the combination (Student_ID, Course_ID) can. Thus, a composite key plays a vital role in maintaining data integrity, preventing duplicate entries, and defining relationships between entities in a relational database..



Relationship

Relationship refers to the logical connection or association between two or more tables based on data they share. It is used to link records from different tables, ensuring that the database reflects real-world connections among entities. Relationships are established through keys—most commonly by linking a primary key in one table to a foreign key in another. This connection enables data integrity, eliminates redundancy, and supports efficient data retrieval.

Types of Relationship

One-to-One (1:1): Each record in one table is associated with only one record in another table.
Example: Each passport is assigned to only one citizen.

One-to-Many (1:N): A record in one table can be associated with multiple records in another table, but each record in the second table relates to only one in the first.
Example: One teacher can teach many students, but each student has only one assigned teacher.

Many-to-Many (M:N): Records in one table can be associated with multiple records in another table, and vice versa. This type of relationship is usually implemented using a junction (bridge) table.
Example: Students can enroll in many courses, and each course can have many students.

Data types

A data type determines the type of data that can be stored in a database column. Although many data types are available, three of the most commonly used data types are

Alphanumeric: used to store characters, numbers, special characters, or nearly any combination.

Examples:

1. CHAR(n): Fixed-length character string (e.g., CHAR(10))
2. VARCHAR(n): Variable-length character string (e.g., VARCHAR(50))
3. TEXT / CLOB: Long text data (used for large paragraphs or documents)

Numeric: can only be used to store numeric values, that is, numbers.

Examples:

1. INT / INTEGER: Whole numbers (e.g., 10, -25)
2. FLOAT / REAL: Numbers with decimal points (e.g., 3.14, -0.99)
3. DECIMAL / NUMERIC: Fixed-point numbers with defined precision and scale (e.g., 123.45)

Date and time: Date and time data types are used to store date and time values, which widely vary depending on the relational database management system (RDBMS) being used.

Examples:

1. DATE: Stores calendar dates (e.g., '2025-11-09')
2. TIME: Stores time of day (e.g., '14:30:00')

Database languages

Database languages, also known as query languages or data query languages, are a classification of programming languages that developers use to define and access databases, which are collections of organized data that users can access electronically.

These languages allow users to complete tasks such as controlling access to data, defining and updating data, and searching for information within the database management system (DBMS).

A DBMS provides necessary database languages that allow users to express database updates and queries, which are requests for data. Some examples of database languages are SQL, Oracle, dBase, MS Access, FoxPro, etc.

Database languages are divided into four categories on the basis of the kind of operations they perform –

1. Data Definition Languages (DDL)
2. Data Manipulation Languages (DML)
3. Data Control Language(DCL)
4. Transaction Control Language(TCL)

1. Data Definition Languages (DDL)

It is used to define database structure or pattern. It is used to create schema, tables, indexes, constraints, etc. in the database. Using the DDL statements, you can create the skeleton of the database. Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.The set of statements in DDL used to implement database schema are as follows:

CREATE: It is used to create objects in the database.

ALTER: It is used to alter the structure of the database

DROP: This command is used to delete a relation in the database or an entire database.

TRUNCATE: This command deletes all the entries from the relation but keeps the relation structure secured in the database.

RENAME: This command renames the relation in a database.

COMMENT: It is used to comment on the data dictionary.

2. Data Manipulation Languages (DML)

Data manipulation language (DML) provides operations that handle user requests, offering a way to access and manipulate the data that users store within a database. Its common functions include inserting, updating, and retrieving data from the database. Here's a list of DML statements:

INSERT: Adds new data to the existing database table

UPDATE: Changes or updates values in the table

DELETE: Removes records or rows from the table

SELECT: Retrieves data from the table or multiple tables

MERGE: It performs UPSERT operation, i.e., insert or update operations.

3. Data Control Language (DCL)

Data control language (DCL) controls access to the data that users store within a database. Essentially, this language controls the rights and permissions of the database system. It allows users to grant or revoke privileges to the database. Here's a list of DCL statements:

GRANT: Gives a user access to the database

REVOKE: Removes a user's access to the database

4. Transaction control language (TCL)

Transaction control language (TCL) manages the transactions within a database. Transactions group a set of related tasks into a single, executable task. All the tasks must succeed in order for the transaction to work. Here's a list of TCL statements:

COMMIT: Carries out a transaction

ROLLBACK: Restores a transaction if any tasks fail to execute

SAVEPOINT: Sets a point in a transaction to save