CSE 301 – DATABASE

Lecture 6

Chapter 3: Relational Model

Introduction to Relational Model

- Relational model is the theoretical basis of relational databases.
- The relational model of data based on the concept of relations.
- A "Relation" is a mathematical concepts based on the ideas of sets.
- The Relational Model was proposed by E. F. Codd for IBM in 1970 to model data in the form of relations or tables.

What is Relational Model?

- **Relational model** represents how data is stored in relational databases.
- A relational database stores data in the form of relations (tables).
- After designing the conceptual model of database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDBMS languages.
- RDBMS languages: Oracle, MySQL, SQL etc.

What is RDBMS?

- RDBMS stands for "*Relational Database Management System*".
- RDBMS is basis for SQL and all other modern database system like MySQL Server, IBM DB2, Oracle, MYSQL, Microsoft Access, etc.
- A **Relational Database Management System (RDMBS)** is a database management system which is based on the **relational model** introduced by **E. F. Codd**
- In relational model, data is stored in relations(tables) and is represented in form of tuples(rows).

Codd's Rule for Relational DBMS

- Rule zero:
  - ✓ This rule states that for a system to qualify as an RDBMS, it must be able to manage database entirely through the relational capabilities.
- Rule 1: Information rule
  - ✓ All information (*including metadata*) is to be represented as stored data in cells of tables.
  - ✓ The rows and columns have to be strictly unordered.

Codd's Rule for Relational DBMS

- Rule 2: Guaranteed Access

  ✓Each unique piece of data(atomic value) should be accessible by Table Name + Primary Key (Row) + Attribute (column).

- Rule 3: Systematic treatment of NULL

  ✓ Null has several meanings, it can mean missing data, not applicable or no value. It should be handled consistently.

  ✓ Also, Primary key must not be null, ever. Expression on NULL must give null.

Codd's Rule for Relational DBMS

- Rule 4: Active Online Catalog
  - ✓ Database dictionary (catalog) is the structure description of the complete Database, and it must be stored online.
  - ✓ The Catalog must be governed by same rules as rest of the database. The same query language should be used on catalog as used to query database.

Codd's Rule for Relational DBMS

- Rule 5: Powerful and Well-Structured Language
  - ✓ One well structured language must be there to provide all manners of access to the data stored in the database.
  - ✓ Example: SQL, etc. If the database allows access to the data without the use of this language, then that is a violation.
- Rule 6: View Updation Rule
  - ✓ All the view that are theoretically updatable should be updatable by the system as well.

Codd's Rule for Relational DBMS

- Rule 7: Relational Level Operation
  - ✓ There must be Insert, Delete, Update operations at each level of relations.
  - ✓ Set operation like Union, Intersection and minus should also be supported.

Codd's Rule for Relational DBMS

- Rule 8: Physical Data Independence
  - ✓ The physical storage of data should not matter to the system.
  - ✓ If say, some file supporting table is renamed or moved from one disk to another, it should not affect the application.

Codd's Rule for Relational DBMS

- Rule 9: Logical Data Independence
  - ✓ If there is change in the logical structure(table structures) of the database, the user view of data should not change.
  - ✓ Say, if a table is split into two tables, a new view should give result as the join of the two tables.
  - ✓ This rule is most difficult to satisfy. Rule 7: Relational Level Operation

Codd's Rule for Relational DBMS

- Rule 10: Integrity Independence
  - ✓ The database should be able to enforce its own integrity rather than using other programs.
  - ✓ Key and Check constraints, trigger etc., should be stored in Data Dictionary. This also make RDBMS independent of front-end.

Codd's Rule for Relational DBMS

- Rule 11: Distribution Independence
  - ✓A database should work properly regardless of its distribution across a network.
  - ✓ Even if a database is geographically distributed, with data stored in pieces, the end user should get an impression that it is stored at the same place.
  - ✓ This lays the foundation of distributed database

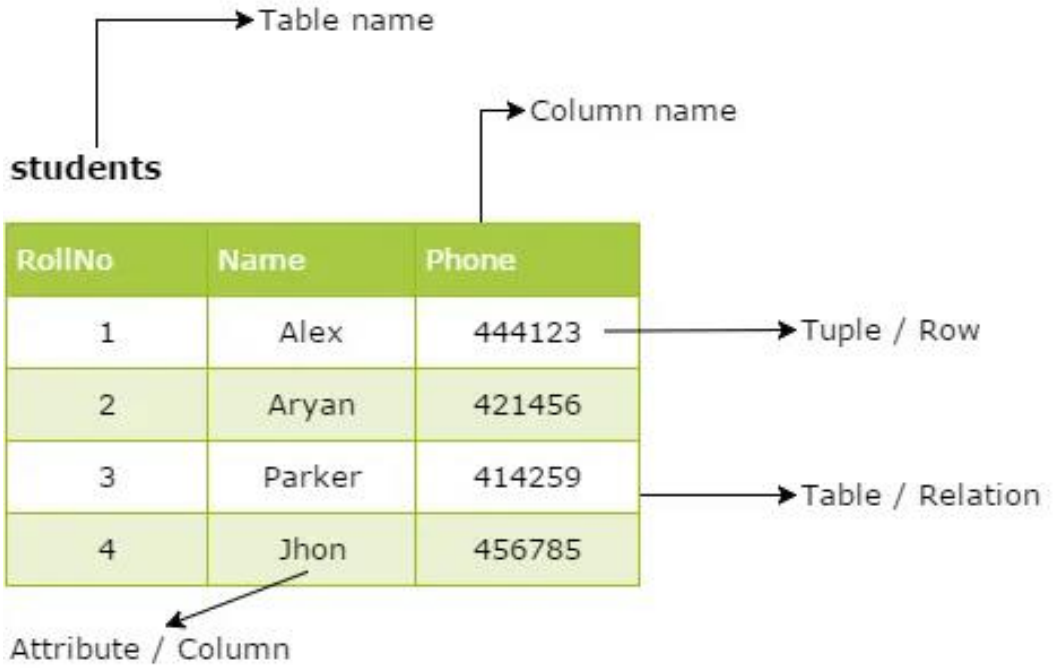
Codd's Rule for Relational DBMS

- Rule 12: Non-subversion Rule
  - ✓ If low level access is allowed to a system, it should not be able to subvert or bypass integrity rules to change the data.
  - ✓ This can be achieved by some sort of looking or encryption

❑ Note: Till now, there is hardly any commercial product that follows all the 12 Codd's rules. Even Oracle follows only eight and half (8.5) out of 12.

Relational Model Concept

- Relational Model can be represented as a table with columns and rows.
- Each row is known as tuple.
- Each column of the table is known as attribute.



Relational Model Terms

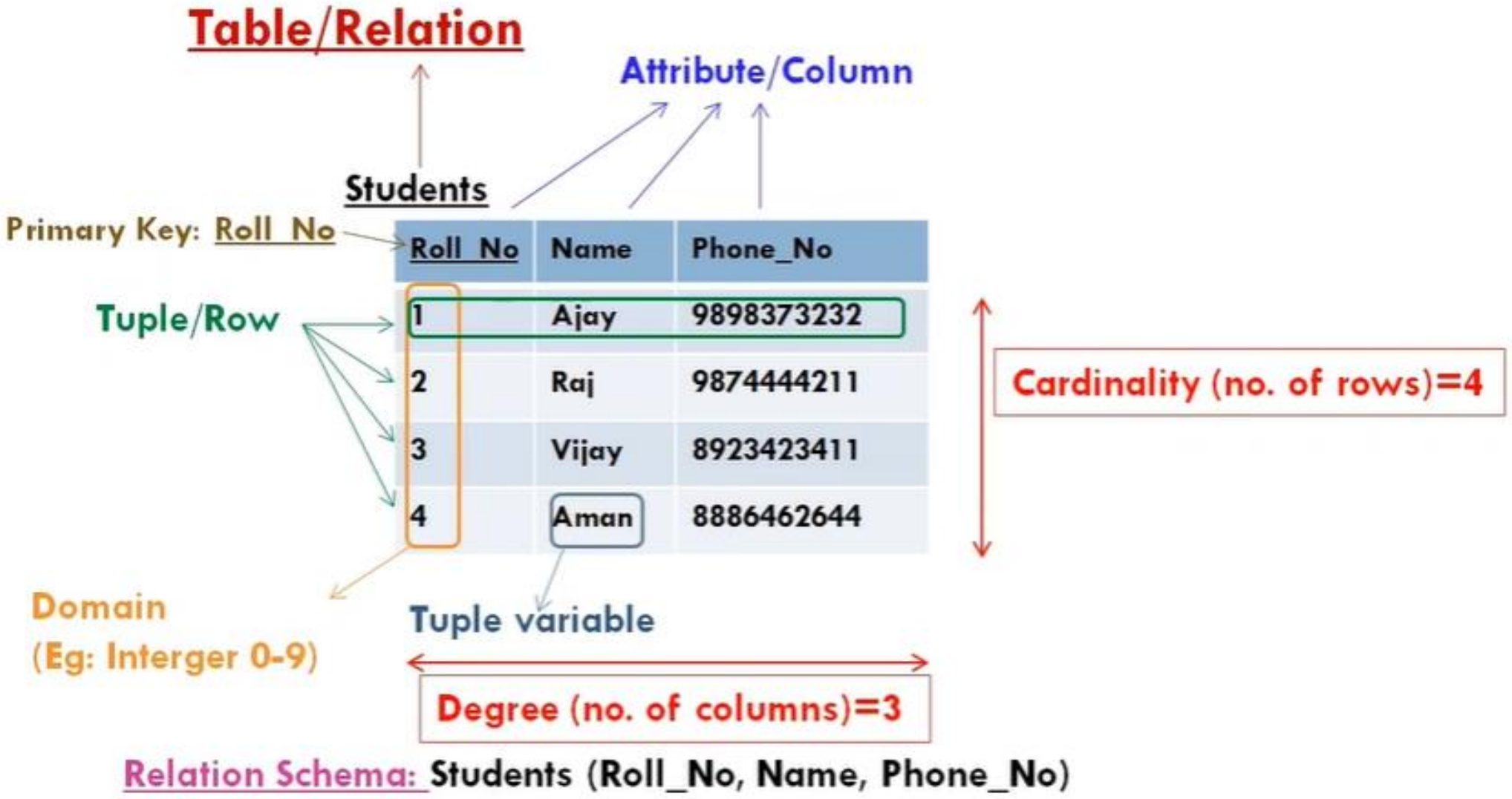
Relational Model Concept

- Relation: A relation is a table with columns and rows.
- Attribute: An attribute is a column of a relation.
- Domain: A domain is the set of allowable values for one or more attributes.
- Tuples: A tuple is a row of relation.
- Relation Schema: A relation schema represents name of the relation with its attributes.
- Relation Instance: It is a finite set of tuples. Relation instances never have duplicate tuples.

Relational Model Concept

- Degree: The total number of columns or attributes present in the relation.
- Cardinality: The total number of rows or tuples present in the relation.
- Relation Key: In a relation, every row has one or multiple attributes. A attribute or set of attributes that can uniquely identify the row in the relation, which is called relation key (or primary key).
- Tuple Variable: It is the data stored in a record in of a table or relation.

Relational Model Concept



Properties of Relational Model

- Each relation has a unique name
- Each row/tuple is unique. No duplicate row.
- Entries in any column have the same domain.
- Each attribute or column has a unique name.
- Order of row or column is irrelevant. i.e. relations are unordered.
- Each cell of relation contains exactly only one value. i.e. attribute values required to be atomic.

Relational Model: Mathematical Structure

- Relation is the relationship between the different types of entitles.
- Relation is defined as a n-ary tuple, its n attributes, a1, a2, …., an. Each attribute ai comes from a domain Di.
- It means a particular relation is formed of n attributes, a1 to an. Each attribute ai comes from a domain Di i.e., ai ϵ Di .
- So, a relation r is a subset of cross or cartesian product of the sets of Di

$$D1 \times D2 \times D3 \times \ldots \times Dn$$

Relational Model: Mathematical Structure

- Suppose we have two sets D1 and D2 where

  D1 = {1, 4} and D2 = {2, 3, 5}

- The cartesian product of these two sets, written as D1 x D2, is set of all ordered pairs such that the first member of set D1 and second member of set D2.

- D1 x D2 = {(1, 2), (1, 3), (1, 5), (4, 2), (4, 3), (4, 5)}

  So, Subset of D1 x D2

  r = {(1, 2), (4, 2)} is a relation over D1 x D2

Attribute Types

- Name: each attribute of a relation has a name.
- Domain: set of allowable values for one attribute is called domain of the attribute.
- Atomic: Attributes values are required to be atomic. Atomic means it can not be subdivided further.
  - ✓ Multivalued and composite attributes are not atomic.
- Null: Null value is a member of every domain.
  - ✓ Null value is used for unknown values and not applicable values.

Relation Schema

- Schema means that is the ***description of the relation***.
- A1, A2, A3, ……, An are attributes.
- R = (A1, A2, A3, ……, An) is a relational schema.

  E.g., Student-schema (Student_name, Student_address, Student_phone)
- A relation is defined over a schema i.e r(R).
- r(R) is a relation r on the relation schema R

  E.g., Student (Student-schema)

Relation Instance

- The current values (*relation instance*) of a relation are specified by a table.
- R = {(Hieu, HCM, 20/02/1993),

  (Phat, Hanoi, 15/09/1997),

  (Dat, Dai An, 07/11/1995),

  (Thu, HCM, 30/06/1991),}

Integrity Constraints

- Integrity Constraints are used to ensure accuracy and consistency of the data in a relational database.
- It is a set of rules that the database not permitted to violate.
- Constraints may apply to each attribute or may they apply to relationships between tables.
- It ensures that changes (update, deletion, insertion) made to database by authorized users do not result in a loss of data consistency.
- Thus, the *integrity constraints guards against accidental damage to the database*.

E.g., A blood group must be ‘A’ or ‘B’ or ‘AB’ or ‘O’ only (can not any other values).

Types of Integrity Constraints

- Domain Constraints
  - ✓ Used in Attributes.
- Entity Integrity Constraints
  - ✓ Used in Attributes.
- Referential Integrity Constraints
  - ✓ Used in Relationship.
- Key Constraints
  - ✓ Used in Attributes.

Domain Constraint

- Domain constraints refers to the rules defined for the valid set of values that can be stored for a certain attribute.
- The data types of domain includes string, character, integer, time, date, etc.
- The value must be available in the corresponding domain.

STUDENT_ID	NAME	SEMESTER	AGE
101	Manish	1st	18
102	Rohit	3rd	19
103	Badal	5th	20
104	Amit	7th	A

Not allowed. Because AGE is an integer value

Entity Integrity Constraints

- It states that primary key value can not be Null.
- This is because primary key value is used to identify individual rows in a relation.
- If primary key value is null, then we can not find those rows.
- A table can contain null value other than primary key field.

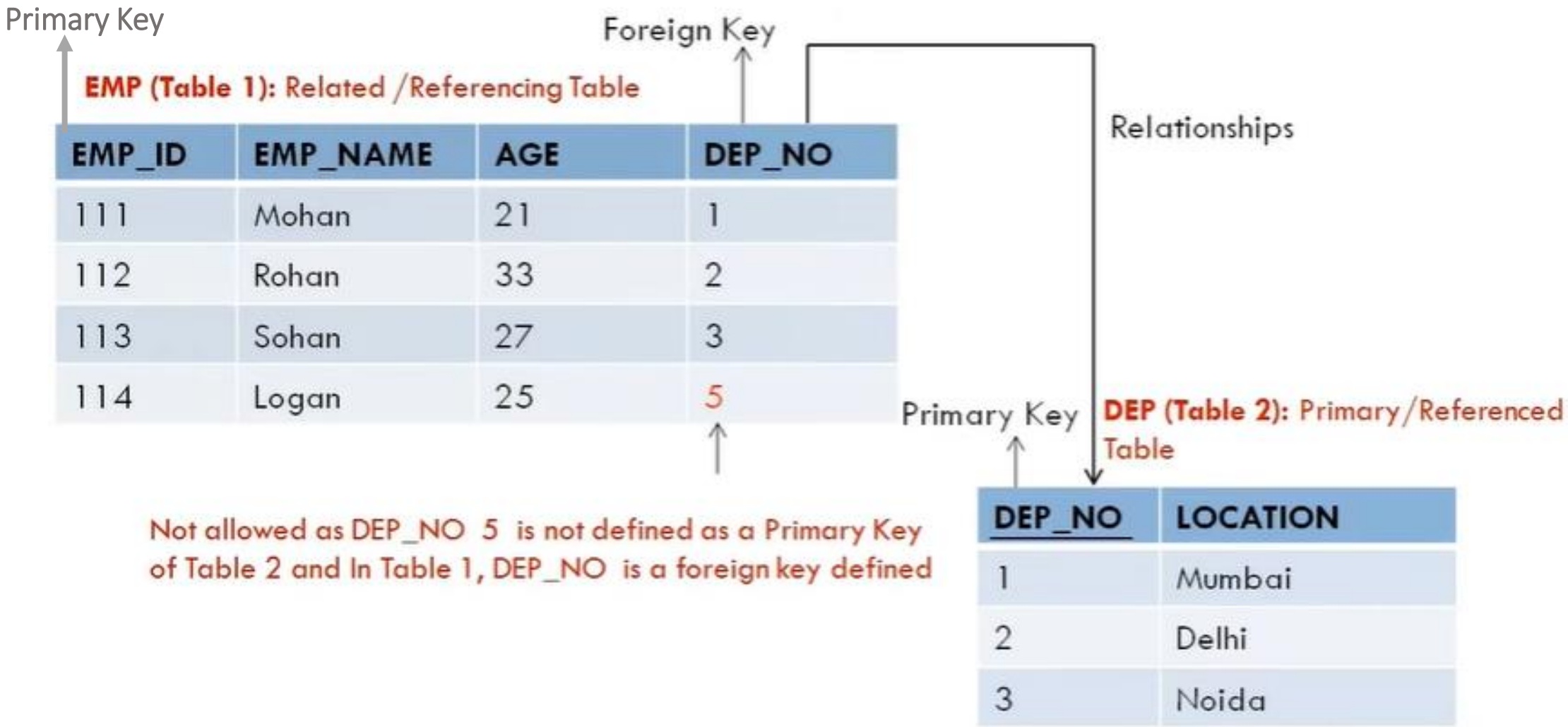
EMP_ID	EMP_NAME	SALARY
111	Mohan	20000
112	Rohan	30000
113	Sohan	35000
	Logan	20000

Not allowed as Primary Key can't contain NULL value

Referential Integrity Constraints

- It is specified between two tables.
- It is enforced when a foreign key references the primary key of a table.
- In Referential Integrity Constraints, if a foreign key of Table 1 refers to the primary key of Table 2, then either every value of foreign key in Table 1 must be available in primary key value of Table 2 or it must be null.
- There are some rules present for Referential Integrity Constraints.

Referential Integrity Constraints



EMP_ID	EMP_NAME	AGE	DEP_NO
111	Mohan	21	1
112	Rohan	33	2
113	Sohan	27	3
114	Logan	25	5

DEP_NO	LOCATION
1	Mumbai
2	Delhi
3	Noida

Referential Integrity Constraints

- The rules are:
  - ✓ You can not delete a record from a primary key table if matching records exist in related tables.
  - ✓ You can not change a primary key value in the primary key table if the record exist in related tables.
  - ✓ You can not insert a value in foreign key field of the related table that does not exist in primary key of the primary key table.
  - ✓ However, you can enter a Null value in the foreign key, specifying that the records are unrelated.

Key Constraints

- An entity set can have multiple keys or candidate keys (minimal super key), but out of which one will be the primary key.
- Key Constraints specifies that in any relation
  - ✓ All the values of the primary key should be unique.
  - ✓ The value of the primary must not be Null.

STUDENT_ID	NAME	SEMESTER	AGE
101	Manish	1st	18
102	Rohit	3rd	19
103	Badal	5th	20
102	Amit	7th	21

Not allowed. Because all rows must be **unique**

Relational Query Language

- Relational database system are expected to be equipped with a query language that assist its user to query the database instances.
- Query language is a language which user request information from the database. E.g., ***SQL***.
- Query = "Retrieval Program"
- Two types of Query language:
  - ✓ Procedural Query Language
  - ✓ Non-procedural Query Language

Procedural Query Language

- Procedural Query Language
  - ✓ User instructs the system to perform a series of operations to produce the desire results.
  - ✓ User tells *what data to be retrieve from the database and how to retrieve it.*
- Non-procedural (or Declarative) Query Language
- ✓ User instructs the system to produce the desire results without telling the step-by-step process.
- ✓ User tells *what data to be retrieve from the database but does not tell how to retrieve it*.

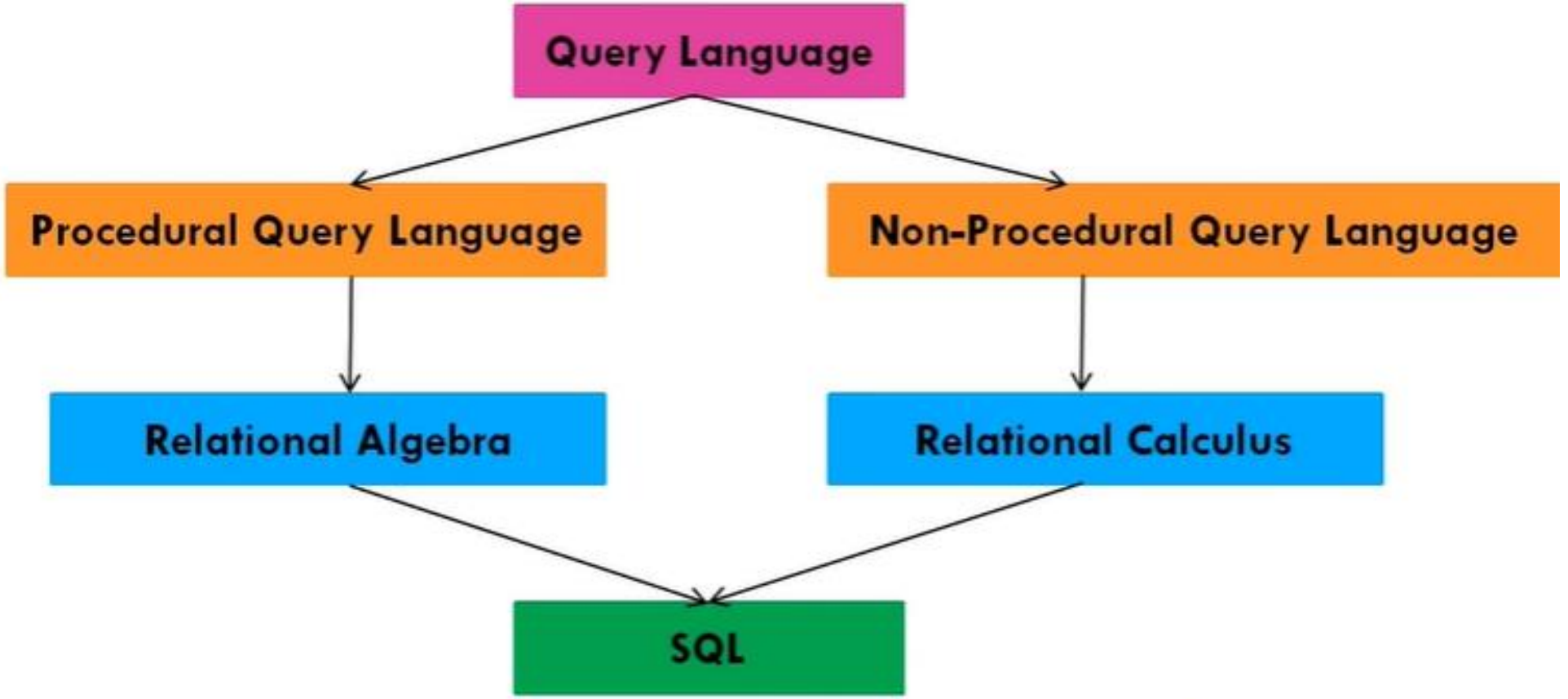
Two “Pure” Query Languages

- These are the theoretical concepts based on mathematical representation.
- Two “Pure” Query Languages or Two “Mathematical ” Query Languages:
  - ✓ Relational Algebra
  - ✓ Relational Calculus
    - ➢ Tuple Relational Calculus
    - ➢ Domain Relational Calculus

Relational Algebra vs Relational Calculus

- Relational Algebra
  - ✓ It is a procedural query language.
  - ✓ It is more operational, very useful for representing execution plan.
  - ✓ Procedural: What data is required and how to get those data.
- Relational Calculus
  - ✓ It is a Non-procedural query language.
  - ✓ It is non-operational or declarative.
  - ✓ Non-procedural: What data they want without specifying how to get those data.

Relational Algebra vs Relational Calculus



Relational Algebra, Calculus, RDBMS & SQL

Relational Model is a **theoretical concept.**

RDBMS is a **practical implementation** of **relational model.**

SQL (Structured Query Language) is **used to write query** on **RDBMS**

Relational algebra and calculus are the **Mathematical system** or **Query language** used on **relational model.**

- Understanding *Relational Algebra and Calculus* is key to understand ***SQL Query Language***

SQL is a **practical implementation** of **relational algebra and calculus.**

Relation Model	RDBMS
Relational Algebra, Relational Calculus	SQL
Algorithm	Code
Conceptual	Reality
Theoretical	Practical

Relational Algebra

- It is a procedural query language which takes a relation as an input and generates a relation as an output.
- It is a language for expressing relational database queries.
- It uses operators to perform queries.
- An operator can binary or unary.
- Types of operators/operations in Relational Algebra
  - ✓ 1. Basic / Fundamental Operators
  - ✓ 2. Additional / Derived Operators
- It operations work on one or more relations to define another relation without changing the original relation.
- Relation algebra operations are performed recursively on a relation.

Relational Algebra

- In a relation algebra, input is a relation (or table from which data has to be accessed) and output is also a relation (a temporary table holding the data asked by the users.)
- Relational Algebra works on whole table at one, so we do not have to use loops to interact over the all rows or tuples of the data.

We can use relational algebra To fetch data from this table

Id	Name	Age
1	Hieu	25
2	Phat	21
3	Dat	24
4	Tu	22

Select the names of students under the age of 23.

Name
Phat
Tu

The o/p of the query is also a table or relation with results in different columns

Relational Algebra Operations

- Select Operation (σ)
- Project Operation (∏)
- Union Operation (∪)
- Set Difference (-)
- Cartesian Product (X)
- Rename Operation (ρ)

- ✓ Select, project and rename are unary operator because they operate on one relation.
- ✓ Union, set difference, cartesian product are binary operator because they operate on two relation.

- Natural Join (⋈)
- Left Outer Join (⟕)
- Right Outer Join (⟖)
- Full Outer Join (⟗)
- Set Intersection (∩)
- Division (÷)
- Assignment (←)

- ❖ All are binary operator because they operate on two relation.