- **Integrity Constraint** : Integrity constraints are a set of rules. It is used to maintain the quality of information.

→ Integrity constraints ensure that the data insertion, deletion, updation and other processes have to be performed in such a way that data integrity is not affected.

- Types of Integrity Constraints :

1) **Domain Constraints** : It can be defined as the definition of a valid set of values for an attribute. The value of the attribute must be available in the corresponding domain only.

e.g:

Admission_no.	Name
302	ABC
3014	PQR
A	XYZ

X violation of Domain constraint because admission_no. is an integer attribute, not allowed character value.

2) Referential Integrity Constraints:

It is specified between two tables.

primary key : A field/attribute that can uniquely identifies tuple/value.

foreign key : A field/attribute that refers to the primary key in another table.

This constraint states that insertion is not possible in child table until and unless values of P.K matches with F.K, deletion is possible in child table. But In parent table deletion is not possible as it is referred by child table. However, insertion of new tuples in Parent table is allowed.

1) In Parent table / Primary key:

deletion ✗

insertion ✓

2) In Child table / Foreign key:

deletion ✓

insertion ✗

From RIC we can conclude following:

1) No. of values in

Primary key = Foreign key ✓ possible

2) No. of values in

Primary key > Foreign key ✓ possible

3) No. of values in

Primary key < Foreign key ✗

3) Entity integrity constraints:

The entity integrity constraint states that primary key value can't be null. Because P.K is used to identify individual roces/tuples in relation and if p.k has a null value then we can't find roce/tuples.

NOTE: A table can contain a null value other than the p.k field.

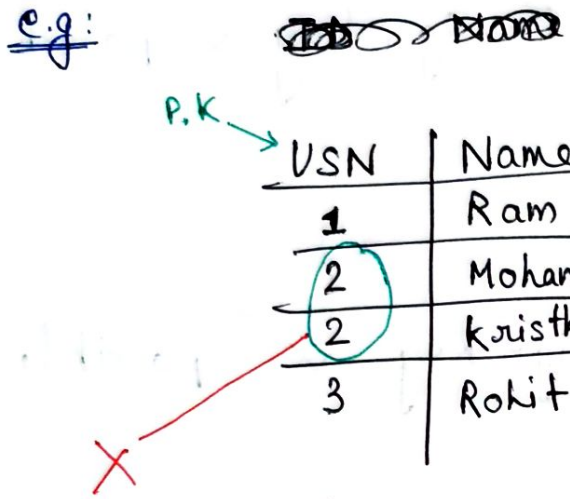
e.g:

ID	Name	Marks
111	ABC	80
NULL	PQR	90
112	WXY	100

✗ not allowed

4) **Key Constraints** : Primary key can contain only unique value in the relational table.

e.g:



not allowed as we can't identify uniquely 2 is for Mohan or it is of kristhma.

Query language:

→ After designing a database, that is ER diagram followed by relational model. Now next task is how to store, retrieve and modify data in the database.

→ Query language is a language using which user request some information from the database.

→ There can be two types:

① Procedural Query language/non-declarative:

Here user instruct the system to perform a sequence of operations on the database, in order to compute the desired result, means user provides both what data to be retrieved and how data to be retrieved.

e.g: Relational Algebra

② Non-Procedural Query language/declarative:

In non-procedural language, the user describes the desired information without giving a specific procedure for obtaining that information.

e.g: Relational Calculus

- Relational Algebra (procedural) and Relational Calculus (non-procedural) are mathematical system/query languages which are used for query on relational model. RA and RC are not executed in any computer they provide the fundamental mathematics on which SQL is based.

- SQL (Structured Query language) works on RDBMS, and it includes elements of both procedural and non-procedural query language, But before going to SQL we need to lear RA and RC.

Relational model	RDBMS
RA, RC	SQL
Algorithm	Code
Conceptual	Reality
Theoretical	Practical

- Relational Algebra :

→ Relational Algebra is one of the two formal query languages associated with relational model. The other one is Relational Calculus.

→ R.A like any other mathematical system provides a number of operations and use relations (tables) as operand.

→ Every operator in the algebra accepts (one/two) relation/table as input arguments and returns always a single relation instance as the result without a name.

→ * It also does not consider duplicacy by default as it is based on set theory. Same query ~~is~~ written in RA and SQL may give different result as SQL considers duplication.

→ Each RA query describes a step-by-step procedure for computing the desired answer, based on the order in which operators are applied in the query. Hence, Relational Algebra is a Procedural Query language / Non-Declarative.

→ Relational Algebra provides the framework for query optimization as it is pure mathematics no use of english keywords, operators are represented as symbol.

P.T.O

- Operators used in Relational Algebra:

(1) Basic / ~~Fund~~ fundamental Operators

Name	Symbol
Select	σ
Project	π
Union	$\cup$
set-difference	$-$
Cross Product	$\times$
Rename	ρ

(2) Derived Operators: To make query more efficient, more optimized, easy to use and easy to understand these operators are derived from fundamental Operators.

Name	Symbol	Derived from
Join	$\bowtie$	$\times$
Intersection	$\cap$	$-$
Division	$\div$	$(\times, -, \pi)$
Assignment	$=$	

- **Relational Schema:** A relational schema R, denoted by $R(A_1, A_2, \ldots A_n)$ is made up of a relation name R and a list of attributes $A_1, A_2, \ldots A_n$. Each attribute A_i is the name of a role played by some domain D in the relation schema R. It is used to describe a Relation.

  e.g: schema representation of table student as:

  STUDENT (NAME, ID, CITY)

- **Relational Instance:** Relations with its data at particular instant of time.

- **The Select Operation:** (Horizontal Selection)

  → The select operation selects tuples that satisfy a given predicate / condition.

  → Lowercase Greek letter sigma (σ) is used to denote selection.

  → Predicate / condition appears as a subscript to σ.

  → ~~Minimum number of tuples~~

P. T. O

→ The argument relation (table) is in parentheses after the σ.

→ $\sigma_{\text{condition}}(\text{table\_name})$

→ Minimum number of tuples selected can be 0. and Maximum selected tuples can be all.

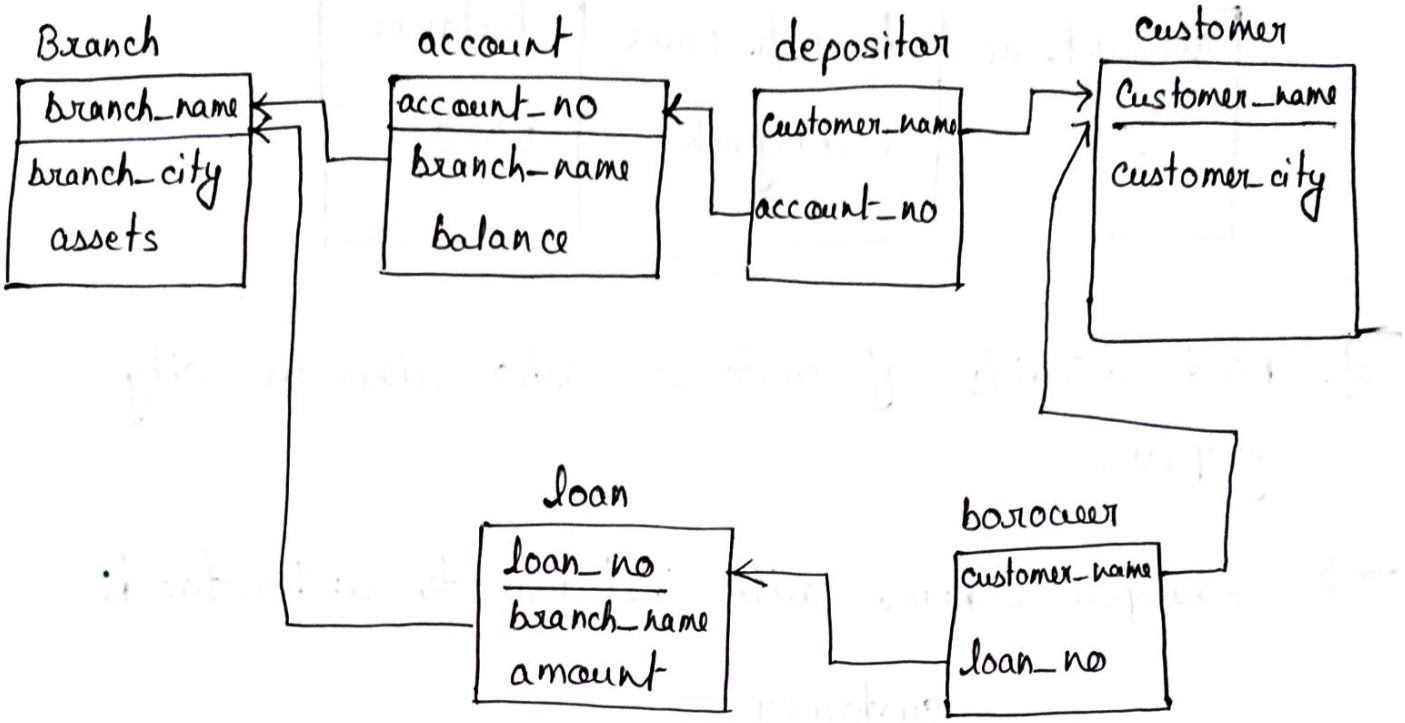
→ It is a unary operator

→ Eliminates only tuples/rows.

→ Degree (Result relation) = degree (parent relation), where degree refers to no. of attributes.

→ 0 <= cardinality (result relation) <= cardinality (parent relation), where cardinality refers to no. of rows/tuples.

Q. Consider the following schema and answer the following questions:



1) Find ^ those accounts where account balance greater than 1000.

details

→ Example schema ^ to understand: with instance

account

account_no	branch_name	balance
1	Bishalgarh	5000
2	Ramnagar	1000
3	Kailash	2000

RA Query:

$$\sigma_{balance > 1000}(account)$$

Ans

Resultant table:

account_no	branch_name	balance
1	Bisalgarh	5000

Q. Find details of customer who lives in city jaipur.

→ Example schema with instance to understand:

customer

customer_name	customer_city
ABC	Bangalore
XYZ	Jaipur
PQR	Jaipur

RA Query:

$$\sigma_{\text{customer\_city}}(\text{customer})$$

Ans

Resultant table:

customer_name	customer_city
XYZ	Jaipur
PQR	Jaipur

Q. Find the details of those loans where branch_name is north delhi and amount is greater than 10,000.

→ Example schema with instance to understand:

loan

loan_no	branch_name	amount
1	North Delhi	5000
2	Kolkata	10,000
3	North Delhi	10,000
4	North Delhi	20,000

RA Query:

$$\sigma_{branch\_name = \text{'New Delhi'} \wedge amount \geq 10000}(loan)$$

Ans

Resultant table:

loan_no	branch_name	amount
4	North Delhi	20,000

Some points to remember:

1. We allow comparison using $=$, $\neq$, $<$, $>$, $\leq$ and $\geq$ in the selection predicate/condition
2. Using the connections and ($\wedge$), or ($\vee$), and not ($\neg$) we can combine several predicates into a large predicate.

• Informal design guidelines for relational schema:

1) Semantics of the Attributes:

Semantics means meaning. Whenever we are going to form relational schema there should be some meaning among the attributes. This semantics relates one attribute to another with some relation.

e.g:

USN	Name	Sem

2) Reducing the Redundant Value in Tuples:

Mixing attributes of multiple entities may cause redandancy. i.e., combining two or more tables.

Problems with update anamolies:

1) Insertion anomalies
2) Deletion anomalies
3) Modification anomalies.

e.g:

Student

USN	Name	Sem

Department

Course	ID

USN	Name	Course	ID

Null

3) Reducing the Redundant Values in Tuples:

Null values may exist due to following reasons:

(i) Missing Data: Sometimes data is missing or unavailable at the time of entry, which results in null values. This can happen due to incomplete user input, data collection errors or system issues.

e.g: if a user hasn't provided their birthdate, the corresponding field could contain a null value.

(ii) Optional Attributes: In cases where certain attributes are optional, null values can indicate that the attribute is not applicable or not provided for a particular record.

To Reduce null values in Tuples, set default values for columns that can have a reasonable default, so that when a value isn't provided, the default value is used instead of null.

4) Disallowing spurious Tuples:

The "lossless join" property is used to guarantee meaningful results for join operations.

- The relations should be designed to satisfy the lossless join condition. No spurious tuples should be generated by doing a natural join of any relation.