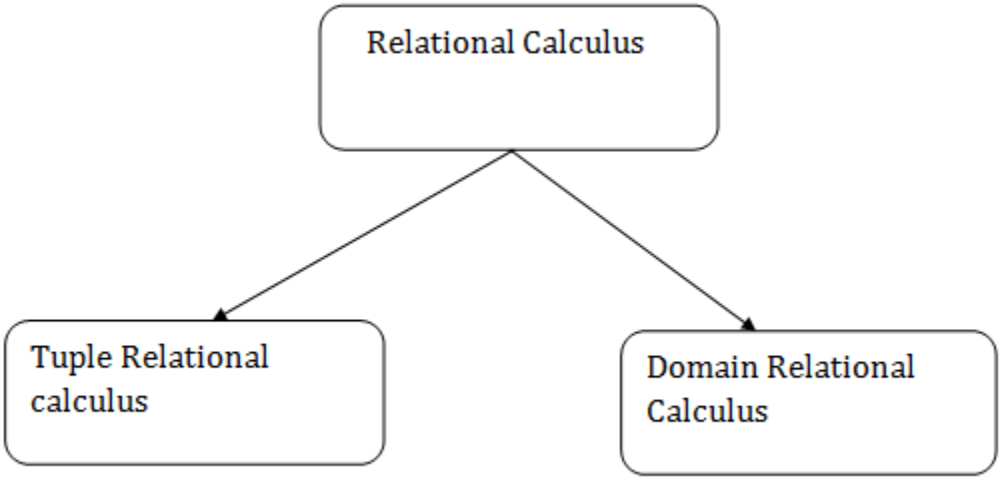
Relational Calculus

- Relational calculus is a non-procedural query language. In the non-procedural query language, the user is concerned with the details of how to obtain the end results.
- The relational calculus tells what to do but never explains how to do.

Types of Relational calculus:



1. Tuple Relational Calculus (TRC)

- The tuple relational calculus is specified to select the tuples in a relation. In TRC, filtering variable uses the tuples of a relation.
- The result of the relation can have one or more tuples.

Notation:

1. {T | P (T)} or {T | Condition (T)}

Where

T is the resulting tuples

P(T) is the condition used to fetch T.

For example:

1. { T.name | Author(T) AND T.article = 'database' }

OUTPUT: This query selects the tuples from the AUTHOR relation. It returns a tuple with 'name' from Author who has written an article on 'database'.

TRC (tuple relation calculus) can be quantified. In TRC, we can use Existential (∃) and Universal Quantifiers (∀).

For example:

1. { R| ∃T ∈ Authors(T.article='database' AND R.name=T.name)}

Output: This query will yield the same result as the previous one.

2. Domain Relational Calculus (DRC)

- The second form of relation is known as Domain relational calculus. In domain relational calculus, filtering variable uses the domain of attributes.
- Domain relational calculus uses the same operators as tuple calculus. It uses logical connectives ∧ (and), ∨ (or) and ¬ (not).
- It uses Existential (∃) and Universal Quantifiers (∀) to bind the variable.

Notation:

1. { a1, a2, a3, ..., an | P (a1, a2, a3, ... ,an)}

Where

a1, a2 are attributes
P stands for formula built by inner attributes

For example:

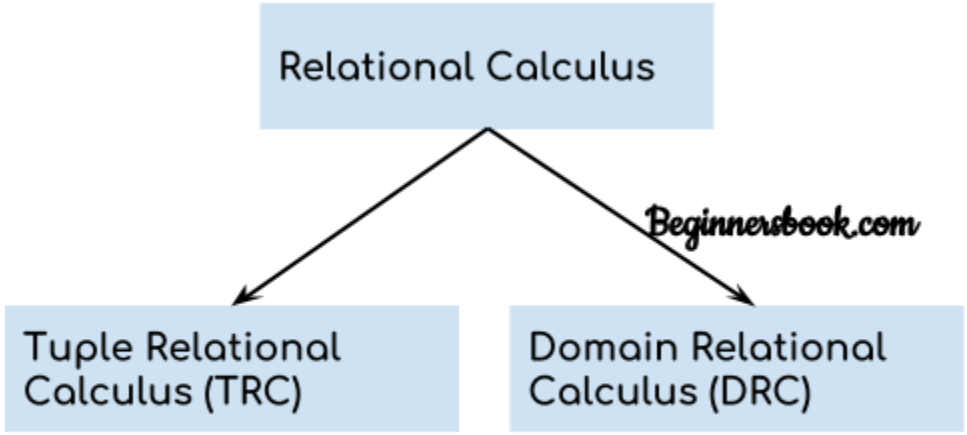
1. {< article, page, subject > | ∈ javatpoint ∧ subject = 'database'}

Output: This query will yield the article, page, and subject from the relational javatpoint, where the subject is a database.

What is Relational Calculus?

Relational calculus is a non-procedural query language that tells the system what data to be retrieved but doesn't tell how to retrieve it.

Types of Relational Calculus



1. Tuple Relational Calculus (TRC)

In tuple relational calculus, we work on filtering tuples based on the given condition.

Syntax: { T | Condition }

In this form of relational calculus, we define a tuple variable, specify the table(relation) name in which the tuple is to be searched for, along with a condition.

We can also specify column name using a . dot operator, with the tuple variable to only get a certain attribute(column) in result.

A tuple variable is nothing but a name, can be anything, generally we use a single alphabet for this, so let's say `T` is a tuple variable.

To specify the name of the relation(table) in which we want to look for data, we do the following:

`Relation(T)`, where `T` is our tuple variable.

For example if our table is **Student**, we would put it as `Student(T)`

Then comes the condition part, to specify a condition applicable for a particluar attribute(column), we can use the `.` dot variable with the tuple variable to specify it, like in table **Student**, if we want to get data for students with age greater than 17, then, we can write it as,

`T.age > 17`, where `T` is our tuple variable.

Putting it all together, if we want to use Tuple Relational Calculus to fetch names of students, from table **Student**, with age greater than **17**, then, for `T` being our tuple variable,

`T.name | Student(T) AND T.age > 17`

For Example:

Table: Student

```
First_Name      Last_Name      Age
----------      ---------      ----
Ajeet           Singh          30
Chaitanya       Singh          31
Rajeev          Bhatia         27
Carl            Pratap         28
```

Lets write relational calculus queries.

Query to display the last name of those students where age is greater than 30

```
{ t.Last_Name | Student(t) AND t.age > 30 }
```

In the above query you can see two parts separated by | symbol. The second part is where we define the condition and in the first part we specify the fields which we want to display for the selected tuples.

The result of the above query would be:

```
Last_Name
---------
Singh
```

Query to display all the details of students where Last name is ‘Singh’

```
{ t | Student(t) AND t.Last_Name = 'Singh' }
```

Output:

```
First_Name     Last_Name      Age
----------     ---------      ----
Ajeet          Singh          30
Chaitanya      Singh          31
```

2. Domain Relational Calculus (DRC)

In domain relational calculus, filtering is done based on the domain of the attributes and not based on the tuple values.

Syntax: $\{ c1, c2, c3, ..., cn \mid F(c1, c2, c3, ... ,cn)\}$

where, c1, c2... etc represents domain of attributes(columns) and F defines the formula including the condition for fetching the data.

For example,

$\{< name, age > \mid \in Student \land age > 17\}$

Again, the above query will return the names and ages of the students in the table **Student** who are older than 17.

In domain relational calculus the records are filtered based on the domains. Again we take the same table to understand how DRC works.

Table: Student

```
First_Name     Last_Name     Age
----------     ---------     ----
Ajeet          Singh         30
Chaitanya      Singh         31
Rajeev         Bhatia        27
Carl           Pratap        28
```

Query to find the first name and age of students where student age is greater than 27

```
{< First_Name, Age > | ∈ Student ∧ Age > 27}
```

Note:
The symbols used for logical operators are: ∧ for AND, ∨ for OR and ¬ for NOT.

Output:

```
First_Name     Age
----------     ----
Ajeet          30
Chaitanya      31
Carl           28
```