DBMS: Relational Calculus

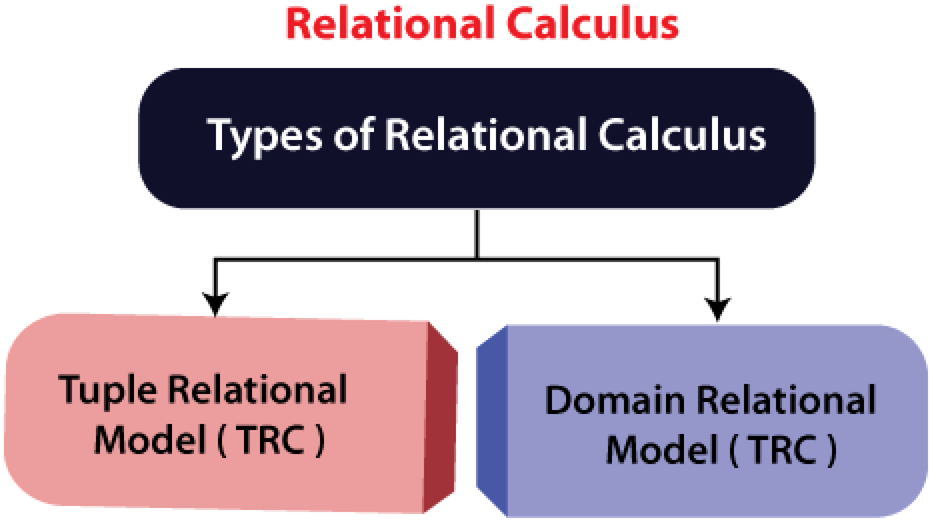
Akhilesh Arya

Relational calculus

- There is an alternate way of formulating queries known as Relational Calculus.
- Relational calculus is a non-procedural query language.
- The relational calculus tells what to do but never explains how to do
- It is based on Predicate calculus, a name derived from branch of symbolic language

- Calculus expressions involves the use of Quantifiers.
- There are two types of quantifiers:
  - **Universal Quantifiers:** The universal quantifier denoted by ∀ is read as for all which means that in a given set of tuples exactly all tuples satisfy a given condition. (for all)
  - **Existential Quantifiers:** The existential quantifier denoted by ∃ is read as for all which means that in a given set of tuples there is at least one occurrences whose value satisfy a given condition. (there exist)

- Types of Relational calculus:


Relational Calculus
Types of Relational Calculus
Tuple Relational Model (TRC)
Domain Relational Model (TRC)

Tuple Relational Calculus (TRC)

- It is a non-procedural query language which is based on finding a number of tuple variables also known as range variable for which predicate holds true.
- 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.

- Syntax:

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

  - T is the resulting tuples
  - P(T) is the condition used to fetch T

Various operators that we can use to create conditions or formula are:

And (∧)

Or (∨)

Not (¬)

Example:

- Consider a relation:
- Author:

ID	Name	Dept_name	Article	Salary

- Write a query in TRC to find the ID, Name, Dept_name of the author who wrote an article on database

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

Cont..

- Previous example can be written using the quantifiers also
- Suppose we need only the names of the writers from the author relation who wrote article on database then

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

We read this expression as: "The set of all the tuples R such that there exist a tuple T in relation Author for which the value of R and T for the name attribute are equal, and the value of T for the article is equal to database

Cont..

- Consider the same relation of the previous example and write a TRC query to find the ID of the author with a salary is greater than 80k

$$\{ t \mid \exists s \in \text{Authors}\ (t[ID] = s[ID] \ \land \ s.salary>80000)$$

Cont..

- Consider one more relation Department

Dept_ID	Dept_name	Building

- Write a TRC query to find the names of all the authors whose department is in i3 labs

Cont..

$$\{\, t \mid \exists s \in \text{Authors}\ (t[\text{name}] = s[\text{name}]\ \land \\ \exists u \in \text{Department}\ (u[\text{Dept\_name}] = s[\text{Dept\_name}]\ \land \\ u[\text{building}] = \text{“i3”}))\}$$

- Find sid’s of all the students who have taken all courses offered by the department computer science
- Relations are
    - Student (sid, name, age)
    - Course (cid, cname, dept_name)
    - Takes (sid, cid)

Cont..

$$\begin{array}{l}\{t \mid \exists\, r \in \text{student}\ (t[\text{sid}] = r[\text{sid}] \ \land \\ (\forall\, u \in \text{course}\ (u[\text{dept\_name}] = \text{‘Biology’} \rightarrow \\ \exists\, s \in \text{takes}\ (u[\text{cid}] = s[\text{cid}] \ \land s[\text{sid}]=t[\text{sid}]))))\}\end{array}$$

Customer

Customer name	Street	City
Saurabh	A7	Patiala
Mehak	B6	Jalandhar
Sumiti	D9	Ludhiana
Ria	A5	Patiala

Account

Account number	Branch name	Balance
1111	ABC	50000
1112	DEF	10000
1113	GHI	9000
1114	ABC	7000

Branch

Branch name	Branch city
ABC	Patiala
DEF	Ludhiana
GHI	Jalandhar

Loan

Loan number	Branch name	Amount
L33	ABC	10000
L35	DEF	15000
L49	GHI	9000
L98	DEF	65000

Borrower

Customer name	Loan number
Saurabh	L33
Mehak	L49
Ria	L98

Depositor

Customer name	Account number
Saurabh	1111
Mehak	1113
Sumiti	1114

Queries:

1. Find the loan number, branch, amount of loans of greater than or equal to 10000 amount.
2. Find the loan number for each loan of an amount greater or equal to 10000.
3. Find the names of all customers who have a loan and an account at the bank.
4. Find the names of all customers having a loan at the “ABC” branch.

Answers:

- {t| t ∈ loan ∧ t[amount]>=10000}

- {t| ∃ s ∈ loan(t[loan number] = s[loan number] ∧ s[amount]>=10000)}

- {t | ∃ s ∈ borrower(t[customer-name] = s[customer-name]) ∧ ∃ u ∈ depositor(t[customer-name] = u[customer-name])}

- {t | ∃ s ∈ borrower(t[customer-name] = s[customer-name] ∧ ∃ u ∈ loan(u[branch-name] = “ABC” ∧ u[loan-number] = s[loan-number]))}

Domain Relational Calculus (DRC)

- In domain relational calculus, filtering variable uses the domain of attributes.
- Domain relational calculus uses the same operators as tuple calculus.
- The QBE or Query by example is a query language related to domain relational calculus.

Cont..

- Syntax:

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

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

Example:

- Consider following relations:
  - instructor (id, name, dept_name, salary)
- Find the instructor id, name , dept_name, salary for instructor whose salary is greater than 80000

$$\{<i,n,d,s> \mid <i,n,d,s> \in \text{instructor} \land \text{salary} > 80000\}$$

Example:

- Consider following relations:
  - Student (sid, name, age)
  - Course (cid, cname, dept_name)
  - Teaches (sid, cid)
  - instructor (tid, name, dept_name, salary)

Customer

Customer name	Street	City
Debomit	Kadamtala	Alipurduar
Sayantan	Udaypur	Balurghat
Soumya	Nutanchati	Bankura
Ritu	Juhu	Mumbai

Loan

Loan number	Branch name	Amount
L01	Main	200
L03	Main	150
L10	Sub	90
L08	Main	60

Borrower

Customer name	Loan number
Ritu	L01
Debomit	L08
Soumya	L03

Queries:

- Find the loan number, branch, amount of loans of greater than or equal to 100 amount.

- Find the loan number for each loan of an amount greater or equal to 150.

- Find the names of all customers having a loan at the “Main” branch and find the loan amount .

Answer:

- $\{\prec l, b, a\succ \mid \prec l, b, a\succ \in loan \land (a \geq 100)\}$

- $\{\prec l\succ \mid \exists\ b, a\ (\prec l, b, a\succ \in loan \land (a \geq 150)\}$

- $\{\prec c, a\succ \mid \exists\ l\ (\prec c, l\succ \in borrower \land \exists\ b\ (\prec l, b, a\succ \in loan \land (b = \text{“Main”})))\}$