RELATIONAL MODEL

Ongole Campus

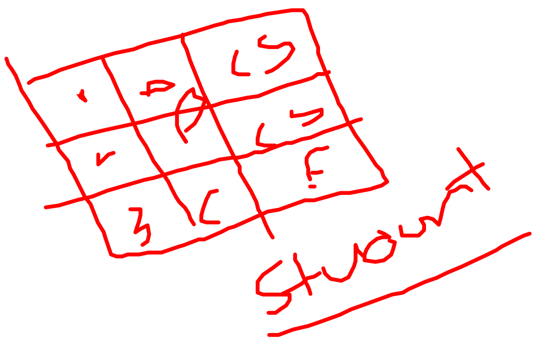
Rajiv Gandhi University of Knowledge Technologies

Relational Model

- It represents as data base collection of relations.
- Relation – Table of values.
- A row in table represents related data values.
- A row represents entity/relationship in real world.

Relational Model

- The table name and column names interpret the meaning of the values in each row.
- All values in a column are of the same data type.



Relational Model

Terminology:

- Table – Relation.
- Column header – Attribute.
- Row – Tuple.
- Domain – The data type, describing the type of values that can appear in each column.

Relational Model

Relation schema:

Let,

R is a relation name and $A_1, A_2, A_3, \ldots, A_n$ are its attributes.

Then the relation schema is denoted as

$$R(A_1, A_2, A_3, \ldots, A_n)$$

Relational Model

Relation schema:

Example:

Student (id, name, dept, year, semester, contact)

Here,

Student is Relation name.

id, name, dept, year, semester, contact are attributes.

Relational Model

Relation schema with data types:

- A relational schema can be specified using its data type as follows –

  relation_name (Attribute: data_type, ...)

- Example:

  Student (id : integer, name : string, dept : string, semester : integer, year : integer, contact : integer)

Relational Model

Domain:

- Set of ***atomic*** values.
- Atomic – Each value in the domain is indivisible in the domain.

Relational Model

Domain:

Let,

$R(A_1, A_2, A_3, \ldots , A_n)$ denotes a relation schema.

D is domain of the an attribute A_i.

Then the domain is represented as

$$\boldsymbol{Dom}(A_i)$$

Relational Model

Domain:

Example:

Student(id : integer, name : string, dept : string, contact : integer)

Dom(id) = Student id numbers.

Dom(name) = Names.

Dom(dept) = Departments in the institute.

Dom(contact) = Phone numbers in India.

Relational Model

Degree of relation:

- The degree (or arity) of a relation is the number of attributes **n** of its relation schema.
- Example:

If Student(id, name, dept, semester, year, contact) is given relation schema then its degree is ***"6".***

Relational Model

Characteristics of relation:

- **Ordering of tuples in a relation –**
  - Tuples in a relation need not have any particular order.

Relational Model

Characteristics of relation:

- Ordering of tuples in a relation – Example

Id number	Name	Dept
1234	Sathya	CSE
1235	Radha	ECE
1236	Raju	CIVIL

Id number	Name	Dept
1235	Radha	ECE
1236	Raju	CIVIL
1234	Sathya	CSE

Relational Model

Characteristics of relation:

- **Ordering of Values within a Tuple and an Alternative Definition of a Relation –**
  - The order of attributes in tuples is not important.
  - An alternative definition to the relation can be given by changing the positions of the attributes in the relation.

Relational Model

Characteristics of relation:

- **Ordering of Values within a Tuple and an Alternative Definition of a Relation – Example**

((id,1234), (name,sathya), (dept,CSE))

((dept,CSE), (id,1234), (name,sathya))

Relational Model

Characteristics of relation:

- **Values and NULLs in the Tuples –**
  - Each value in a tuple is an **atomic value.**
  - composite and multi-valued attributes are not allowed.
  - NULL value is used to represent
    - The values of attributes that may be unknown.
    - The values of attributes may not apply to a tuple.

Relational Model

Characteristics of relation:

- **Interpretation (Meaning) of a Relation –**
  - The relation schema can be interpreted as a declaration or a type of assertion.

Relational Model

Relation instance:

- Refers the relation at a specific instance.
- The tuples contains at a particular instance.

Relational Model

Relational model constraints:

- **Domain constraints.**
- **Integrity constraints.**
  - **Key constraints.**
  - **Foreign key constraints.**
- **General constraints.**
  - **Table constraints.**
  - **Assertions.**

Relational Model

Domain constraints:

The domain of a field is essentially the *type of that field, in* programming language terms, and restricts the values that can appear in the field.

Relational Model

Integrity constraints:

- Condition specified on a database schema and restricts the data that can be stored in an instance of the database.
- If a database instance satisfies all the integrity constraints specified on the database schema, it is a legal instance.
- Of course domain constraint is also a kind of integrity constraint.

Relational Model

Key constraints:

- A **key constraint** is a statement that a certain *minimal subset of the fields of a relation is a* unique identifier for a tuple.
- Candidate key.
- Primary key.

Relational Model

Foreign key constraints:

- To establish the interconnection between two tables.
- Referenced relation – the relation which references the other relation.
- Referencing relation - the relation that is referred by referenced relation.
- Foreign key – key in referencing relation that is primary key of referenced relation.

Relational Model

Table constraints:

- Table constraints are associated with a single table and checked whenever that table is modified.

Relational Model

Assertions:

- Assertions involve several tables and are checked whenever any of these tables is modified.

ER diagrams to relational design

Relational Query Language

- A query language is a language in which a user requests information from the database.
- Higher level than that of a standard programming language.
- Provides fundamental techniques to extract the data from database.

Relational Query Language

Types – 2

- Procedural languages.
- Non – procedural languages.

Relational Query Language

Procedural languages:

The user instructs the system to perform a sequence of operations on the database to compute the desired result.

Example: Relational algebra.

Relational Query Language

Non - Procedural languages:

The user describes the desired information without giving a specific procedure for obtaining that information.

Example: Relational calculus.

Relational Algebra

The ***relational algebra*** consists of a set of operations that take one or two relations as input and produce a new relation as their result.

Basic operations –

- Selection (σ) – Selecting rows.
- Projection (Π) – Selecting attributes.

Relational Algebra

Basic operations –

- Selection (σ) – Selecting rows.
- Projection (Π) – Selecting attributes.

- These operations allow us to manipulate data in a single relation.

Relational Algebra

Select all the student details who has the GRADE 'C'

Example: STUDENT

ID	NAME	CONTACT	CGPA	GRADE
1234	RADHA	9000000456	8.2	B
1235	SYAM	8000000987	7.8	C
1236	RAM	7090909023	9.4	X
1237	SHARMA	6784567888	8.9	A
1238	SURABHI	7056078091	7.2	C
1239	SAKHI	9034567890	8.1	B

Relational Algebra

Select all the student details who has the GRADE 'C'

$$\sigma_{GRADE='C'}(STUDENT)$$

ID	NAME	CONTACT	CGPA	GRADE
1235	SYAM	8000000987	7.8	C
1238	SURABHI	7056078091	7.2	C

Relational Algebra

In general, the selection condition is a Boolean combination (i.e., an expression using the logical connectives ∩ and U) of terms that have the form *attribute **op** constant or attribute1 **op** attribute2, where **op** is one of the comparison* operators <, <=, =, ,#, >=, or >.

Relational Algebra

Select all the student names and their grades.

Example: STUDENT

ID	NAME	CONTACT	CGPA	GRADE
1234	RADHA	9000000456	8.2	B
1235	SYAM	8000000987	7.8	C
1236	RAM	7090909023	9.4	X
1237	SHARMA	6784567888	8.9	A
1238	SURABHI	7056078091	7.2	C
1239	SAKHI	9034567890	8.1	B

Relational Algebra

Resultant relation:

Select all the student names and their grades.

$\Pi_{NAME,GRADE}$ (STUDENT)

NAME	GRADE
RADHA	B
SYAM	C
RAM	X
SHARMA	A
SURABHI	C
SAKHI	B

Relational Algebra

Example: STUDENT

ID	NAME	CONTACT	CGPA	GRADE
1234	RADHA	9000000456	8.2	B
1235	SYAM	8000000987	7.8	C
1236	RAM	7090909023	9.4	X
1237	SHARMA	6784567888	8.9	A
1238	SURABHI	7056078091	7.2	C
1239	SAKHI	9034567890	8.1	B

Select the student names and contact numbers whose grade is ‘C’.

Relational Algebra

Step 1: Select the tuples who has GRADE 'C'.

$$\sigma_{GRADE='C'}(STUDENT)$$

ID	NAME	CONTACT	CGPA	GRADE
1235	SYAM	8000000987	7.8	C
1238	SURABHI	7056078091	7.2	C

Relational Algebra

Step 2: Select the attributes names and contact.

$$\Pi_{NAME,GRADE}\ (\sigma_{GRADE='C'}\ (STUDENT))$$

NAME	CONTACT
SYAM	8000000987
SURABHI	7056078091

Relational Algebra

Set operations:

- Union.
- Intersection.
- Difference.
- Cross product.

Relational Algebra

Union (U):

Let A, B are two union compatible relations.

Then A U B,

Returns a relation instance containing all tuples that occur in *either relation instance A or relation instance B (or both)* with the identical schema to A.

Relational Algebra

Union (A U B):

Union Compatibility –

- Two relations have same number of attributes.
- Corresponding attributes have same domains.

Relational Algebra

Intersection (∩):

Let A, B are two union compatible relations.

Then A ∩ B,

Returns a relation instance containing all tuples that occur in *both relations* with the identical schema to A.

Relational Algebra

Difference (–) :

Let A, B are two union compatible relations.

Then A – B,

Returns a relation instance containing all tuples that occur in *A but not in B* with the identical schema to A.

Relational Algebra

Cross product / Cartesian product (X):

Let A, B are relations.

Then A X B,

- Returns a relation instance with the schema that contains all attributes A and B.
- The result of *A X B* contains all the tuple *(1, s) (the concatenation of tuples A and B) for each pair* of tuples *l E A, s E B.*

Relational Algebra

Other operations:

- Renaming.
- Join.
- Division.

Relational Algebra

Renaming (ρ) –

- Rename the given relation or attributes names.
- $\rho_{\text{new\_name}}$ (relation_name).
- $\rho_{\text{new\_name(List of attributes)}}$ (relation_name).

Relational Algebra

Join (⋈) –

- Binary operation.
- Combine information from two or more relations.
- A join can be defined as a cross-product followed by selection operation.

Relational Algebra

Join (⋈) –

Types:

- Natural join.
- Theta join.
- Equi join.

Relational Algebra

Natural Join (⋈)

Joining of two relations *A and B, denoted A* $\bowtie$ *B, in* which we pair only those tuples from *A and A that agree in whatever attributes* are common to the schemas of *A and B.*

Relational Algebra

Natural Join (⋈)

Let, A and B are two relation schemas in which C_1, C_2, . . . C_n are common attributes in A and B.

Then, the resultant relation schema consist the attributes of A and B where the common attributes appears only one time.

The resultant relation tuples are the combined tuples of A and B where the values of common tuples are equal.

Relational Algebra

Natural Join (⋈)

Example:

Student:

ID	NAME	DEPT
1234	RADHA	CSE
1235	KRISHNA	ECE
1236	SHANTHI	CSE

Marks:

ID	SUB-1	SUB-2	SUB-3
1234	78	89	88
1235	56	95	70
1237	90	97	96

Relational Algebra

Natural Join ($\bowtie$)

Example:

Result

ID	NAME	DEPT	SUB-1	SUB-2	SUB-3
1234	RADHA	CSE	78	89	88
1235	KRISHNA	ECE	56	95	70

Relational Algebra

Theta Join ($\bowtie_c$)

It is also known as ***conditional join***.

Joining of two relations *A and B, denoted as A* $\bowtie_c$ *B, in* which we pair only those tuples from *A and A that agree the given condition.*

Relational Algebra

Theta Join ($\bowtie_c$)

The resultant relation schema consist of all the attributes of given two relations.

Relational Algebra

Equi Join ($\bowtie_c$)

It is special case theta join in which the condition do the equality check.

Relational Algebra

Division (/) –

Let A(Z) / B(X) where the attributes of *B are a subset of the attributes of A; that is,* $X \subseteq Z$*. Let Y be the set of* attributes of *A that are not attributes of B; that is,* $Y = Z - X$ *(and hence* $Z = X \cup Y$*).*

Then,

Division Produces a relation *R(Y) that includes all tuples t[X] in A(Z) that appear in A in combination* with every tuple from *B(X),*

Relational Algebra

Basic operations

- Selection.
- Projection.
- Union.
- Difference.
- Cartesian product.
- Rename

Derived operations

- Intersection.
- Join.
- Division.

Relational Algebra

Intersection:

- Intersection can be derived in terms of ***difference***.

$$\mathbf{A \cap B = A - (A - B)}$$

Relational Algebra

Question:

List out the subject names and ids that are taught in both semester 1 and 2 in the year e1?

subjects(id, name, dept, sem, year)

Relational Algebra

Question:

What is the largest salary among the employs ?

emp(id, name, dept, salary)

$$\Pi_{salary}(emp) - \Pi_{salary}(emp \bowtie_{(emp.salary < e.salary)} \rho_{e}(emp))$$

Relational Algebra

Extended operations:

- Generalized projection.
- Aggregation.

Relational Algebra

Generalized projection –

- Extends the projection operation by allowing arithmetic and string operations.
- $\Pi_{F1,F2,F2,\ldots,FN}$ (RELATION)
- F1, F2, F3, ... , FN are algebraic expressions.

Relational Algebra

Generalized projection –

- The expressions are may contains arithmetic operations.
- Example: $\Pi_{id,name,salary+1000}$ (Employ)

Relational Algebra

Aggregation (G) –

- Aggregate operation permits the use of aggregate function.
- An aggregate function take a collection of values and return a single value.

Relational Algebra

Aggregation (G) –

- Example functions:
  - Sum
  - Avg
  - Count
  - Max
  - Min
  - Count-distinct

Relational Algebra

Aggregation (G) –

- Example : Count number of persons participating in the games.

  Games(id, name, category, game)

$$G_{count\text{-}distinct(id)}(Games)$$

Relational Algebra

Aggregation (G) –

General form:

$$_{G_1,G_2,G_3....,G_N}\mathcal{G}_{F_1(A_1),F_2(A_2)....F_N(A_N)}\text{ (Relation)}$$

$$\mathcal{G}_{count\text{-}distinct(id)}\text{(Games)}$$

Relational Algebra

Aggregation (G) –

Example:

Find the count of students in each department.

Students(id, name, dept, contact)

$${}_{dept}G_{count(id)}(Students)$$

Relational Algebra

Writing relational algebra expressions for queries:

- Find the list of employee id numbers who has salary above 30000.

  Emp(id, name, dept, salary)

Relational Algebra

Writing relational algebra expressions for queries:

- Find the list of employee id numbers and names who has salary above 30000 and below 60000.

  Emp(id, name, dept, salary)

Relational Algebra

Writing relational algebra expressions for queries:

- Find the list of dept that has less than 10 employees.

  Emp(id, name, dept, salary)

Relational Algebra

Writing relational algebra expressions for queries:

- Find the list of id, name, dept of the students who has grade ‘A’.

  Students(id, name, dept, contact)

  Marks(id, percent, cgpa, grade)

Relational Calculus

The ***relational calculus*** uses predicate logic to define the result desired without giving any specific algebraic procedure for obtaining that result.

Types:

- Tuple relational calculus.
- Domain relational calculus.

Relational Calculus

Introduction to tuple relational calculus:

- Expression is as follows.

$$\{t \mid P(t)\}$$

Where, t is tuples, P(t) is the formula that consist of several tuple variable. P(t) should be true for ‘t’.

The tuple variable should be quantified by ∃ or ∀.

Relational Calculus

Introduction to tuple relational calculus:

Formula is built out of following atoms:

- s ϵ R, where ‘s’ is tuple variable and ‘R’ is a relation.
- s[x] Θ u[y], where ‘s’ & ‘u’ are two tuple variables and x & y are attribute on which ‘s’ & ‘u’ are defined. Θ is comparison operator that are <, >, =, ≠, ≤, ≥.

Relational Calculus

Introduction to tuple relational calculus:

Formula is built out of following atoms:

- s[x] Θ c, where 's' is a tuple variables and x is an attribute on which 's' is defined, c is a constant. Θ is comparison operator that are $<$, $>$, $=$, $\neq$, $\leq$, $\geq$.

Relational Calculus

Introduction to tuple relational calculus:

We can combine multiple formulae atoms using following rules:

- If P_1 is formula then, we can write as $\neg P_1$ or (P_1).
- If P_1 and P_2 are formulae then, we can write as
  - $P_1 \wedge P_2$
  - $P_1 \vee P_2$
  - $P_1 \rightarrow P_2$

Relational Calculus

Introduction to tuple relational calculus:

We can combine multiple formulae atoms using following rules:

- If P_1 (s) is formula containing free variable 's' and relation 'R', then we can write the formula as follows.
  - $\exists s \in R\ (P_1(s))$
  - $\forall s \in R\ (P_1(s))$

Relational Calculus

Writing basic queries using tuple calculus:

- Find the employee details who has salary above 30000.

  Emp(id, name, dept, salary)

$$\{t \mid t \in Emp \wedge t[salary] > 30000\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find employee details who has salary above 30000 and below 50000.

  Emp(id, name, dept, salary)

$$\{t \mid t \in Emp \wedge t[salary] > 30000 \wedge t[salary] < 50000\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find employee details who are not belongs to "ECE" department .

  Emp(id, name, dept, salary)

$$\{t \mid t \in Emp \wedge \neg(t[dept] = \text{“Ece”})\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find employee details who are belongs to either "CSE" or "ECE" department .

  Emp(id, name, dept, salary)

$$\{t \mid t \in Emp \wedge (t[dept] = \text{“ECE”} \vee t[dept] = \text{“CSE”})\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find employee ids those who has salary above 30000 .

  Emp(id, name, dept, salary)

$$\{t \mid \exists s \in Emp (s[salary] > 30000 \wedge t[id] = s[id])\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find employee ids those who has salary above 30000 and below 50000 .

  Emp(id, name, dept, salary)

$\{t \mid \exists s \in Emp\ (\ s[salary] > 30000 \wedge (\ s[salary] > 50000 \wedge t[id] = s[id])\}$

Relational Calculus

Writing basic queries using tuple calculus:

- Find the student names who has grade 'A' .

  Students(id, name, dept, contact)

  Results(id, Percent, CGPA, grade)

$$\{t \mid \exists s \in Students\ (\ (\ \exists r \in Results\ (r[id] = s[id] \land r[grade] = \text{'A'}) \land\ t[name] = s[name]\ )\ )\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find the all student ids that is taught in both sem–1 and sem-2 of year ‘E1’.

  Subjects(id, name, sem, year)

$$\{t \mid \exists s \in Subjects\ (\ t[id] = s[id] \wedge s[sem]=1 \wedge s[year] = \text{‘E1’})\}$$

$$\wedge$$

$$\{t \mid \exists s \in Subjects\ (\ t[id] = s[id] \wedge s[sem]=2 \wedge s[year] = \text{‘E1’})\}$$

Relational Calculus

Writing basic queries using tuple calculus:

- Find the all student ids and names, who took all the courses that were offered by department of “CSE”.

  Subjects(s_id, s_name, s_dept, contact)

  Course(c_id, c_name, c_dpt,)

  Takes(s_id, c_id)

  $\{t \mid \exists s \in \text{Students} (t[id] = s[id] \wedge t[name]=s[name] \wedge$

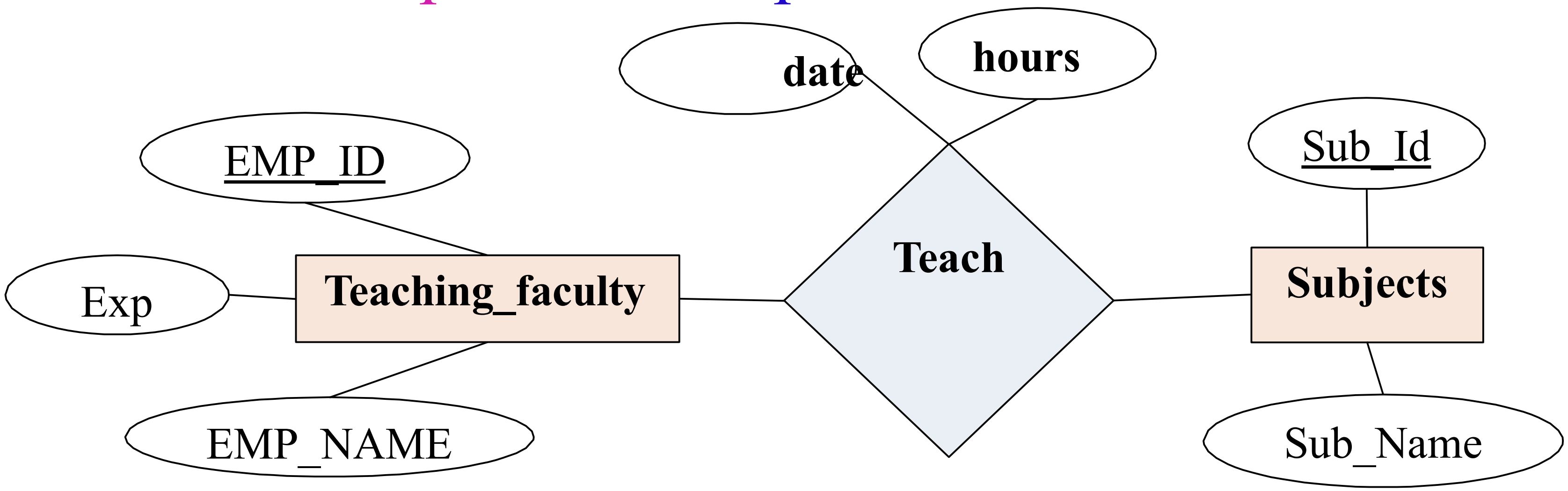
  $(\forall c \in \text{Course} (c[c\_dept] = \text{“CSE”} \rightarrow$

  $\exists a \in \text{Takes} (t[id] = c[s\_id])$

  $\wedge\ c[c\_id] = a[c\text{-}id])) \}$

ER diagrams to Relational Model

Relationship sets: Example



Teaching_faculty(EMP_ID, EMP_NAME, Exp)

Subjects(Sub_Id, Sub_Name, Unique(Sub_name))

Teach(EMP_ID, Sub_Id, date, hours, foreign key(Emp_id),foreign key(Sub_id))

Thank You