Relational Model

- behind every db is a data model
- behind relational db is relational model
- data → table of values or flat file of records
- A row/a record → describes an entity/association of entities
- Each col has col name → table name & col name help to interpret the data

STUDENT

ROLL	NAME	DOB

DEPT

DCODE	DNAME	HOD

STUDENT_DEPT

ROLL	DCODE
1	D1

STUDENT & DEPT db rows represent entities while rows of STUDENT-DEPT db represent association b/w entities of other 2 db

Formal correspondence

Row → Tupple
Column → Attribute
Table → Relation

Each attribute has a domain

Domain is a set of atomic values

Relation Schema of a Relation

$R \rightarrow$ name of relation

$R(A1, A2 \ldots An)$ is a relation schema with R as reln. name & $A1, A2 \ldots An$ are the attributes. A_i is the name of the role played by its corresponding domain D_i ($DOM(A_i)$).

Intension of a relation $\Rightarrow$ Schema

Degree of reln $\rightarrow$ no. of attr. in the schema $\rightarrow n$

Reln or state of a reln, of schema $R(A_1, A_2 \ldots A_n)$ is denoted as $r(R)$ is a set of n-tuples $\{t_1, t_2 \ldots t_m\}_n$ (each tuple contains n values) where t_i is an ordered list of values $\langle v_1, v_2 \ldots v_n \rangle$. v_j belongs to $DOM(A_j)$

R

	A_1	A_2	....	A_n
t_1	v_1	v_2		v_n
t_2				

State of a relation is called extension of a relation.
It is instance-based & changes with time-stamp.

- Mathematically, r(R) is a collection of n-degree mappings on DOM(A1), DOM(A2) .... DOM(A_n)

$\xrightarrow{}$ n-deg relation schema

$$r(R) \subseteq DOM(A1) \times DOM(A2) \ldots \times DOM(A_n)$$

• Characteristics of relation

- Tuples are unordered i.e. it doesn't assume any order in which the tuples are present only that a set of tuples is there

- Attr. in a tuple may be ordered or unordered
ORDERED → tuple is an ordered list of values
UNORDERED → $r(R)$ is a set of mappings $\{t_1, t_2 \ldots t_m\}$ where t_i is a mapping from R to D. D is union of $DOM(A_1), DOM(A_2) \ldots DOM(A_n)$
i.e. In tuple t_i $\{<A_j, v_j>\}$ set of attribute-value pairs

- Values & null values - attribute value is atomic & for a tuple it is single-valued (i.e. for a tuple, an attribute can have only one value)
Value of an attr. either belongs to its domain or is null VALUE ∈ DOMAIN

any schema that satisfies relational model is of 1st normal form

ROLL	SCORE
1	75
2	85
3	90
4	50
5	0/NULL

→ if this is NULL then avg. $= \frac{300}{4} = 75$

if it is 0 then avg $= \frac{300}{5} = 60$

NULL values ignored for statistical operation

Interpretation of NULL value
→ Not Applicable
→ Applicable but unknown
→ Intentionally not stored

- Constraints in relational model

Model-based constraint
⇒ inherent in the model
→ no 2 tuples are same
→ no ordering of tuple can be assumed
→ attr. in a tuple
  – may be unordered or ordered

Data-Dependency Cond.
- Functional dependency
- Multivalued dependency
  - Testing goodness of

Schema-based constraint
⇒ can be specified at the time of defining the schema (using DDL)

Application-based constraint
→ can't be specified using DDL but using logic within the appl.
→ The special constraints including multiple relations may require application level validations

GUI is event-driven. Normally execution of statements is sequential in order but in GUI, when event occurs, the registered actionListener listens & corresponding method is invoked.

To the listener, the event object is sent denoting the src, thus single listener may handle multiple events

e.g.

ROLL	SCORE

LOW_LIMIT	HIGH_LIMIT	GRADE

This design ensures that discrepency in assigning grade for score is prevented

- <u>Domain constraint</u> (type of Schema constraint)
  – Attribute ∈ its domain

- <u>Key constraint</u>
  S_K be a subset of attributes of a schema R if for any 2 tuple $t_1, t_2 \in r(R)$ such that
  $$t_1[S_K] \neq t_2[S_K]$$
  then S_K is the super key of R
  Worst case: entire tuple constitutes the super key
  – Minimal super key of which no proper subset is a super key is called CANDIDATE KEY

$\sigma_{P1 \text{ AND } P2}(R)$
$\equiv \sigma_{P_1}(\sigma_{P2}(R))$ or
$\sigma_{P_2}(\sigma_{P1}(R))$

→ <u>Project Openn.</u>

$\pi_{ROLL, NAME}(STUDENT)$

$\pi_{List}(R)$

$\pi_{L2}(\pi_{L1}(R))$
$= \pi_{L1}(\pi_{L_2}(R))$
iff $List2 \subseteq List1$

Q. select roll & name of all students with score >=80
$\pi_{ROLL, NAME}(\sigma_{SCORE >= 80}(STUDENT))$

→ <u>Rename Openn.</u> (Assignment)

$TEMP \leftarrow \sigma_{SCORE >= 80}(STUDENT)$
⇑
ASSIGNMENT

$T \leftarrow$ reln. algebra exp.
o/p is assigned to T
$T(A, B, C) \leftarrow$
Assignment involving renaming of attr. to A, B, C

- Notations :
  - For reln. $R(A_1, A_2 \ldots A_n)$
    - → $\rho S(B_1, B_2, \ldots B_n)(R)$: renaming of reln. & attr.
    - → $\rho S(R)$: renaming of reln only
    - → $\rho(B_1, B_2 \ldots B_n)(R)$: renaming of attr. only

→ <u>Cartesian Product</u>
$R_1(X, A, B) \times R_2(X, P, Q)$

→ Equijoin

STUDENT ⋈ DEPT
STUDENTS.DCODE = DEPT.DCODE

→ o/p schema : student & dept relns combined but only those tuples retained for which predicate satisfies

→ equivalent to cartesian prod. + filtering

Fundamental opern.

- select
- projection
- rename
- cartesian prod.
- union
- set min
- intersection

Extension

- equijoin
- natural join

→ Natural join

STUDENT * DEPT

based on the equality of the common attn, → should appear only once in o/p schema

DEPT

DCODE	DNAME
D1	—
D2	—
D3	—

STUDENT

ROLL	NAME	....	DCODE
			D1
			D1
			D2
			NULL
			D3

Now, equijoin

$$\text{DEPT} \bowtie_{\text{DEPT.DCODE} = \text{STUDENT.DCODE}} \text{STUDENT}$$

If joining attr. is NULL → no match

* when data from multiple relations reqd. → go for joining.

If we want those tuple, for which, for the attr under consideration values of one reln. don't match with values of the other go for outer join

- <u>Outer Join</u>

Left outer Join

$$R1 ⟕_{R1.A = R2.A} R2$$

⇓

Tuples of R1 with no match in R2 will be matched with a null tuple of R2

Right Outer Join

$$R1 ⟖ R2$$

For R2 all tuples appear

Full outer Join

$$R1 ⟗ R2$$

All tuples of R1 & R2 appear assuming null tuple of other reln. where necess

• <u>Outer Union</u>

R1 (X, Y) R2 (X, Z)

Tuples of R1 & R2 match if

$t_1[X] = t_2[X]$

$t_1 \in r_1(R_1)$, $t_2 \in r_2(R_2)$

R1 outer union R2

For matched tuples, value of Y taken from R1 & value of Z from R2

Matched pair ⇒ one output tuple

For unmatched tuple ⇒ it takes null for addln attr.

Full outer join & outer union are equivalent if full outer join taken wrt all common attr. and not any specific subset

• <u>Division opn</u>

ATTENDANCE(<u>ROLL SCODE</u>)

SUB(SCODE)

ATTENDANCE ÷ SUB ⇒ ROLL nos for which tuples combine
of SUB are
with an SCODE
in ATTENDANCE

$R(X,Y) = R(Z) \div S(Y)$ $Y \subset Z$, $X = Z - Y$

Output schema : X

in its state, we have set of tuples $\{t\}$, t is such that corresponding to t there are tuples t_R in $r(R)$ with $t[X] = t_R[X]$ & $t_R[Y] = t_S[Y]$ $\forall\, t_S \in r_S(R)$

t will be in output if it appears in $r(R)$ with combination of all t_S in $r_S(S)$

Division opern by sequence of Fundamental opern.

$R(X, Y)$ $S(Y)$

$T1 \leftarrow \pi_X(R) \rightarrow$ duplicate eliminating proj. $\rightarrow$ gives ROLL of student that appeared for atleast 1 sub.

$T2 \leftarrow T1 \times S \rightarrow$ all combinations of roll x subj.

$T3 \leftarrow T2 - R \rightarrow$ missing roll, sodre combn.

$T4 \leftarrow \pi_X(T3) \rightarrow$ roll of stds missed atleast one sub.

$T \leftarrow T1 - T4 \rightarrow$ roll of students who missed no sub.

$T \equiv R \div S$

- To find out students with same score in same subjects (i.e. to determine students who have copied off of each other)

$$\pi_{RESULT.ROLL,\ COPY.ROLL}\left(RESULT \bowtie_{\substack{RESULT.SCORE = COPY.SCORE \\ AND\ RESULT.ROLL < COPY.ROLL}} \rho_{COPY}(RESULT)\right)$$