Database and Information Systems

Course Roadmap

Chapter 1	Introduction to Databases
Chapter 2	Integrity Constraints and ER Model
Chapter 3	Relational Databases and Schema Refinement
Chapter 4	Query Language
Chapter 5	Transaction and Concurrency Control
Chapter 6	Indexing

Introduction to Relational Model

attributes
(or columns)

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

tuples
(or rows)

Faculty Database

Attribute values are (normally) required to be **atomic**; that is, indivisible

Relation Schema and Instance

- $R = (A_1, A_2, \ldots, A_n)$ is a *relation schema*
  - $A_1, A_2, \ldots, A_n$ are *attributes*
  - Example: *instructor* = (*ID, name, dept_name, salary*)
- The current values (**relation instance**) of a relation are specified by a table
- An element ***t*** of ***r*** is a *tuple*, represented by a *row* in a table

Relations are Unordered

- Tuples may be stored in an arbitrary order
- Example: *instructor* relation with unordered tuples

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Keys

- Let $K \subseteq R$
- *K* is a **superkey** of *R* if values for *K* are sufficient to identify a unique tuple of each possible relation *r(R)*
  - Example: {*ID*} and {ID,name} are both superkeys of *instructor.*
- Superkey *K* is a **candidate key** if *K* is minimal
  Example: {*ID*} is a candidate key for *Instructor*
- One of the candidate keys is selected to be the **primary key**.
  - which one?
- **Foreign key** constraint: Value in one relation must appear in another
  - **Referencing** relation
  - **Referenced** relation
  - Example – *dept_name* in *instructor* is a foreign key from *instructor* referencing *department*

Relational Query Languages

- Procedural vs .non-procedural, or declarative
- Relational algebra is a procedural language
  - Consists of 6 basic operations
    - Projection
    - Selection
    - Union
    - Cross Product
    - Rename
    - Set Difference
  - Derived operations
    - Join
    - Intersection
    - Division
  - Each Query input is a table (or set of tables)
  - Each query output is a table

Project Operation – selection of columns (Attributes)

- Projects **column(s)** that satisfy a given predicate
  - Retrieve the data/column
    - Output distinct records in resulting table
- Relation *r*

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

- $\Pi_{A,C}\,(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Select Operation – selection of rows (tuples)

- Selects **tuple(s)** that satisfy the given predicate or condition from a relation
- Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Union of two relations

- Performs union between two given relations
  - **r**, and **s** must have the same number of attributes
  - Attribute domains must be compatible
  - Duplicate tuples are automatically eliminated
- Relations *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cup s$:

A	B
α	1
α	2
β	1
β	3

- Assume, *r* relation is a bank account, *s* relation is a loan information
- $r \cup s$ shows record that have bank account or taken loan or both

Set difference of two relations

- Output tuples, which are present in one relation but are not in the second relation

- Relations *r*, *s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- *r* – *s*
  - Tuples which are present in r but not in s

A	B
α	1
β	1

- Conditions
  - No of attributes must be same
  - Domain of attributes must be compatible

Set intersection of two relations

- Discover all the tuples that are present in both **r** and **s**

- Relation *r, s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cap s$

A	B
α	2

Note: $r \cap s = r - (r - s)$

- Conditions
  - No of attributes must be same
  - Domain of attributes must be compatible

Cartesian-product

- Cartesian product or cross product
  - Combines information of two different relations into one
- Relations *r, s*:

r

A	B
α	1
β	2

s

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

- *r* x *s*:

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

- Total Columns
  - Columns(r) + Columns(s)
- Total Rows
  - Rows(r) * Rows(s)

Cartesian-product

- Cartesian-product
  - What it is?
    - A basic relation algebra operator to combine two relation instances
  - How it is done?
    - Multiply two tables
  - Where we use it?
    - When there are two or more tables and

      we are not able to get result from single table

A	*B*
α	1
β	2

r

C	*D*	*E*
α	10	a
β	10	a
β	20	b
γ	10	b

s

A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Renaming a Table

- Allows us to refer to a relation, (say E) by more than one name.

$$\rho_X(E)$$

returns the expression *E* under the name *X*

Renaming a Table

- Relations *r*

A	B
α	1
β	2

r

- $r \times \rho_s(r)$

r.A	r.B	s.A	s.B
α	1	α	1
α	1	β	2
β	2	α	1
β	2	β	2

- **Question:** Find the maximum value of B
  - $\Pi_{r.B} - \underbrace{\Pi_{r.B}(\sigma_{r.B<s.B}(r \times \rho_s(r)))}_{\text{B values that are lesser than some B values}}$

Division

- Division operator A÷B (or A/B) can be applied if
  - **Attributes of B is proper subset of Attributes of A**
- The relation returned by division operator will return those **tuples from relation A which are associated to every B's tuple**
  - The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)
- Example: A(x,y)/B(y)
  - Results x values for that there should be tuple (x,y) for every y value of relation B

Division

- Division
  - What it is?
    - A derived relation algebra operator
  - Where we use it?
    - All, every
  - How it is done?
    - Using basic operations (next slides)

Division

- The relation returned by division operator will return those **tuples from relation A which are associated to every B's tuple**
  - The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)
- **Example: A(x,y)/B(y)**
  - **Results x values for that there should be tuple (x,y) in A for every y value of relation B**
- **Question:** Retrieve *Sid* of students who enrolled in *every (or all)* course
- E (Sid, Cid) / C (Cid) = S1

Sid	Cid
S1	C1
S2	C1
S1	C2
S3	C2

Enrolled E

Cid
C1
C2

Course C

Division

- The relation returned by division operator will return those **tuples from relation A which are associated to every B's tuple**
  - The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)

- **Retrieve *Sid* of students who enrolled in *every (or all)* course**
  - E(Sid,Cid)/C(Cid)
  - $\Pi_{sid}(E) - \Pi_{Sid}(\underbrace{\Pi_{Sid}(E) \times \Pi_{Cid}(C) - E}_{})$

Students **Sid** who are not enrolled in at least one course: Disqualified Sid

Sid	Cid
S1	C1
S2	C1
S1	C2
S3	C2

Enrolled E

Cid
C1
C2

Course C

Joins

- Joins
  - Used when we need to combine data from various tables

<u>**E_no**</u>	**E_name**	**E_add**
1	Ram	Indore
2	Gopal	Hyderabad
3	Suresh	Lucknow
4	Mohan	Chennai
5	Bharat	Kochi

Table: Employee

<u>**D_no**</u>	**D_name**	**E_no**
D1	CS	1
D2	EE	2
D3	ME	4
D4	HR	5

Table: Department

- Question: Find E_name who belongs to Hyderabad (Use single Table)
- Question: Find E_name who is working in CS Department(Use both the Tables)
  - We will not get result from one Table

Joins

- Question: Find **E_name** who is working in some Department

E_no	E_name	E_add
1	Ram	Indore
2	Gopal	Hyderabad
3	Suresh	Lucknow
4	Mohan	Chennai
5	Bharat	Kochi

Employee Table: E

D_no	D_name	E_no
D1	CS	1
D2	EE	2
D3	ME	4
D4	HR	5

Department Table: D

Joins

- Types of Joins
  - Cross Join
  - Natural Join
  - Self Join
  - Theta Join or Condition Join
  - Equi Join
  - Outer Join

Cross Join

- Cross Join
  - Combine information of two diff. tables into one

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

- **Cross Join**: *r x s*

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

- Composition of Operations
  - Build expressions using multiple operations
  - Example: $\sigma_{A=C}(r \times s)$
- $\sigma_{A=C}(r \times s)$

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

Natural Join

- Let r and s be relations on schemas R and S respectively. Then, the "natural join" of relations R and S is a relation on schema $R \cup S$ obtained as follows:
  - Consider each pair of tuples t_r from r and t_s from s.
  - If t_r and t_s have the same value on each of the attributes in $R \cap S$, add a tuple t to the result, where
    - t has the same value as t_r on r
    - t has the same value as t_s on s
- Perform a Natural Join only if there is at least one common attribute that exists between two relations

Natural Join Example

- Relations r, s:

r

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

s

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ε

- Natural Join
  - $r \bowtie s$

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

$$\Pi_{r.A,\, r.B,\, r.C,\, r.D,\, s.E}\,(\sigma_{r.B = s.B \,\wedge\, r.D = s.D}\,(r \times s)))$$

Natural Join Example

- Class Activity
  - Find Employee Names who are working in some Department

E_no	E_name	E_add
1	Ram	Indore
2	Gopal	Hyderabad
3	Suresh	Lucknow
4	Mohan	Chennai
5	Bharat	Kochi

Table: Emp

D_no	D_name	E_no
D1	CS	1
D2	EE	2
D3	ME	4
D4	HR	5

Table: Dept

Natural Join Example

- Find Employee Names who are working in some Department

<u>**E_no**</u>	**E_name**	**E_add**
1	Ram	Indore
2	Gopal	Hyderabad
3	Suresh	Lucknow
4	Mohan	Chennai
5	Bharat	Kochi

Table: Emp

<u>**D_no**</u>	**D_name**	**E_no**
D1	CS	1
D2	EE	2
D3	ME	4
D4	HR	5

Table: Dept

- $\Pi_{E\_name}\ (\sigma_{Emp.E\_no=Dept.E\_no}\ (Emp\ x\ Dept))$

Theta Join

- Theta join combines tuples from different relations provided they satisfy the theta condition
  - The join condition is denoted by the symbol **θ**
  - Theta join can use **all kinds of comparison operators**
  - Denoted as $r \bowtie_{\theta} s$

Equi Join

- Equi joins combine tables based on matching values in specified columns
  - When Theta join uses only **equality** comparison operator, it is said to be Equi join
  - Need to specify column

Sid	**Name**	**Std**
101	Alex	10
102	Maria	11

Table: Student

Class	**Subject**	**ID**
10	Math	10ma
10	English	10en
11	Music	11mu
11	Sports	11sp

Table: Subject

- Example: Student $\bowtie_{\text{Student.Std = Subject.Class}}$ Subject

Outer Joins

- Theta Join, Equi Join, and Natural Join are called inner joins
  - Includes only those tuples with matching attributes and the rest are discarded in the resulting relation
- Outer Join
  - Include all the tuples from at least one participating relation
- Left Outer Join
  - Gives matching rows (similar as Natural Join) and rows which are in left table but not in right table
- Right Outer Join
  - Gives matching rows (similar as Natural Join) and rows which are in right table but not in left table
- Full Outer Join
  - Left Outer Join ∪ Right Outer Join

Left Outer Join

- Example:

A	B
100	Database
101	Mechanics
102	Electronics

R

A	C
100	Alex
102	Maya
104	Mira

S

$R ⟕ S$

Left Outer Join

- Example:

A	B
100	Database
101	Mechanics
102	Electronics

R

A	C
100	Alex
102	Maya
104	Mira

S

A	B	C
100	Database	Alex
101	**Mechanics**	**null**
102	Electronics	Maya

R ⟕ S

Modifying the Database

- The content of the database may be modified using the following operations
  - Deletion
  - Insertion
  - Updating
- All these operations are expressed using the assignment operator

Insertion

- Insert tuples (rows) into a relation
  - Specify a tuple to be inserted
  - Write a query whose result is a set of tuples to be inserted
- Insertion is expressed in relational algebra by
  - $r \leftarrow r \cup E$
    - Where r is a relation and E is a relational algebra expression
- Example
  - Insert tuple with $1200 in account A-973 at the Perryridge branch.
    - account $\leftarrow$ account $\cup$ {("Perryridge", A-973, 1200$)}

Deletion

- Remove tuples from a relation
- A deletion is expressed in relational algebra by
  - $r \leftarrow r - E$
    - Where r is a relation and E is a relational algebra expression
- Example
  - Delete all account records in the Perryridge branch
    - $account \leftarrow account - \sigma_{branch\text{-}name = \text{“Perryridge”}}(account)$

Updating

- Change a value in a tuple
- Use the generalized projection operator to do this task
  - $r \leftarrow \prod_{F1,\ F2,\ \ldots,\ Fl} (r)$
- Make interest payments by increasing all balances by 5 percent
  - $account \leftarrow \prod_{AN,\ BN,\ BAL\ *\ 1.05} (account)$
    - Where AN, BN and BAL stand for account-number, branch-name and balance, respectively

More Operations and Functions

- Aggregate Functions
  - We can also apply Aggregate functions
    - SUM, MINIMUM, MAXIMUM, AVERAGE, COUNT

Aggregate Functions

- Gives one aggregated value
- Assume the relation EMP has the following tuples:

Name	Office	Department	Salary
Smith	400	CS	45000
Jones	220	Econ	35000
Green	160	Econ	50000
Brown	420	CS	65000
Smith	500	Fin	60000

- Find minimum salary
  - $F_{\text{MIN (Salary)}}$ (EMP)

MIN(Salary)
35000

- Find average salary
  - $F_{\text{AVG (Salary)}}$ (EMP)

AVG(Salary)
51000

- Count names
  - $F_{\text{Count (Name)}}$ (EMP)

Count(Name)
5

Summary of Relational Algebra Operators

Symbol (Name)	Example of Use
σ (Selection)	$\sigma_{\text{salary} >= 85000}(instructor)$ Return rows of the input relation that satisfy the predicate.
Π (Projection)	$\Pi_{ID,\ salary}(instructor)$ Output specified attributes from all rows of the input relation. Remove duplicate tuples from the output.
x (Cartesian Product)	*instructor* x *department* Output pairs of rows from the two input relations that have the same value on all attributes that have the same name.
∪ (Union)	$\Pi_{name}(instructor) \cup \Pi_{name}(student)$ Output the union of tuples from the *two* input relations.
– (Set Difference)	$\Pi_{name}(instructor)$ -- $\Pi_{name}(student)$ Output the set difference of tuples from the two input relations.
⋈ (Natural Join)	*instructor* ⋈ *department* Output pairs of rows from the two input relations that have the same value on all attributes that have the same name.

References

- Silberschatz, Abraham, Henry F. Korth, and Shashank Sudarshan. *Database system concepts*. Vol. 6. New York: McGraw-Hill, 1997.
- Ramez Elmasri, Shamkant B. Navathe. *Fundamentals of Database Systems.* Edition 6. Pearson, 2010.