DATABASE

Öğr. Gör. Evgin GÖÇERİ

8 Ekim 2014

Relational Algebra -1

- There are various operations that are used to manipulate on relations
- There are variety of languages used by RDBMS(Relational DBMS) one of them is Relational Algebra
- Uses the theory of sets

Relational Algebra -2

- Relational Algebra is a procedural language (SQL is a non procedural language)

(*Procedural DMLs* require a user to specify what data is needed and How to get it.

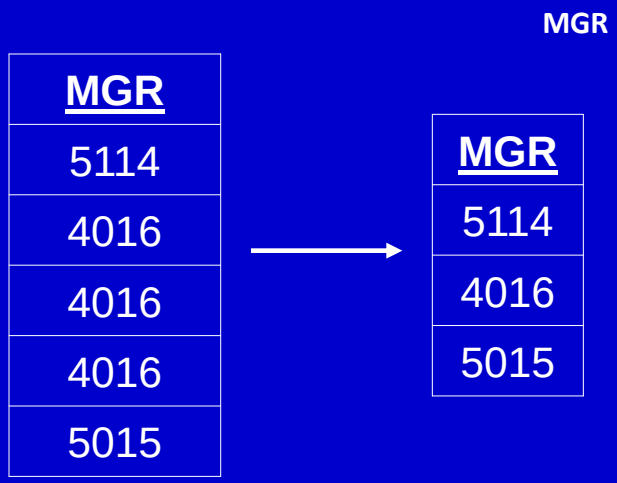
Nonprocedural DMLs require a user to specify what data is needed without specifying How to get it)

Projection

- Selects and outputs a subset of the columns of a relation

- Known as the vertical subsetting operation

EMPLOYEE[MGR] = PJ EMPLOYEE



Selection-1

- Selects the required tuples out of a relation
- Known as the horizontal subsetting operation
- The requirement can be specified by a qualification expression of the form
  “A Θ opr”, where it is a simple predicate expressing a restriction on the values of an attribute represented by A

Selection-2

- “A Θ opr”, A is the attribute name, opr is either an attribute name or a constant and Θ represents one of =, ≠, ≤,<, ≥,>

Selection-3

- Example#1

EMPLOYEE[SALARY≤39000] or

SL EMPLOYEE
SALARY≤39000

2351	J.NORDBY	5114	LANGUAGES	39000
5114	S.AGARWAL	4016	DBSYSTEMS	35000

Selection-4

- Example#2

$$\Pi_{MGR,DEPT}(\sigma_{DEPT=\text{“DBSYSTEMS”}}(EMPLOYEE))$$

(EMPLOYEE [DEPT=“DBSYSTEMS”])[MGR,DEPT]

4016	DBSYSTEMS

Union

- Relations must be union compatible
  - the same degree (number of the attribute)
  - the same column name

EMPLOYEE [ENAME] U EMPLOYEE [DEPT] Ø →

PJ$_{\text{ENAME}}$ EMPLOYEE UN PJ$_{\text{ENAME}}$ SL$_{\text{MGR=4016}}$ EMPLOYEE

J.NORDBY S.AGARWAL G.CANDOR D.SCHRADER K.SMITH	U	S.AGARWAL D.SCHRADER K.SMITH	=	J.NORDBY S.AGARWAL G.CANDOR D.SCHRADER K.SMITH

Difference

- SUPPLIER [SNO] – SUPPLY [SUPPLIER]

or

$$PJ_{SNO}\ SUPPLIER\ \ DF\ \ PJ_{SUPPLIER}\ SUPPLY$$

$$\begin{pmatrix} S1 \\ S2 \\ S3 \end{pmatrix} - \begin{pmatrix} S1 \\ S2 \end{pmatrix} = \begin{pmatrix} S3 \end{pmatrix}$$

Join-1

- The join operation has various names and variants
  - Theta join (‘>’ or ‘<’ is used)
  - Equijoin (‘=’ is used)
  - Natural Join(degree is one less than that of equijoin)

Join - 2

- Example of theta join - Θ

Θ is >

T1=SL EMPLOYEE
DEPT="DBSYSTEMS"

T1 JN EMPLOYEE
SALARY>SALARY

3040	G.CANDOR	4016	DBSYSTEMS	40000	2351	J.NORDBY	5114	LANGUAGES	39000
3040	G.CANDOR	4016	DBSYSTEMS	40000	5114	S.AGAR	4016	DBSYSTEMS	35000
2011	D.SCHRADER	4016	DBSYSTEMS	65000	2351	J.NORDBY	5114	LANGUAGES	39000
2011	D.SCHRADER	4016	DBSYSTEMS	65000	5114	S.AGAR	4016	DBSYSTEMS	35000
2011	D.SCHRADER	4016	DBSYSTEMS	65000	3040	G.CANDOR	4016	DBSYSTEMS	40000

T1

EMPLOYEE

Join - 3

- Example of equijoin Θ is =

SUPPLIER JN SUPPLY
SNO=SUPPLIER

S1	ADAMS	S1	DBSYSTEMS	PEN	100
S1	ADAMS	S1	DBSYSTEMS	PAD	19
S1	ADAMS	S1	LANGUAGES	PAD	8
S2	CLARKE	S2	STORAGESYS	CABINET	2
S2	CLARKE	S2	STORAGESYS	TERMINAL	3

Join -4

- Example of natural join Θ is =
  but dgree is one less than that of equijoin

SUPPLIER NJ SUPPLY
SNO=SUPPLIER

S1	ADAMS	DBSYSTEMS	PEN	100
S1	ADAMS	DBSYSTEMS	PAD	19
S1	ADAMS	LANGUAGES	PAD	8
S2	CLARKE	STORAGESYS	CABINET	2
S2	CLARKE	STORAGESYS	TERMINAL	3

Semijoin-1

- Variation of the join operation

- Two exceptions to join operation
  - No concatenation of tuples in the join
  - Keep the tuple of the relation on the left

Semijoin - 2

- SUPPLIER SJ SUPPLY
  SNO=SUPPLIER

S1	ADAMS
S2	CLARK

Antijoin

- The complement of the semijoin

R AJ S = R DF (R SJ S)

SUPPLIER AJ SUPPLY
SNO=SUPPLIER

S3	SCHWARTZ

Division-1

- Test if a set contains another set

  R DV $S_{c,d}$

R

a	b	c
A	1	C1
A	1	C2
B	2	C2
C	3	C2
C	3	C1

S

d
C1
C2

Partitions of R

A	1	C1
A	1	C2
B	2	C2
C	3	C2
C	3	C1

Division

C1
C2

Answer

A	1
C	3

Division - 2

- Example
  - Which departments are supplied by all items of type A?
  - Hint: all implies division !

T1=PJ (SL ITEM)
INAME TYPE='A'

PEN
PAD

T2=PJ SUPPLY
DEPT,ITEM

DBSYSTEMS	PEN
STORAGESYS	CABINET
STORAGESYS	TERMINAL
DBSYSTEMS	PAD
LANGUAGES	PAD

T3= T2 DV T1

DBSYSTEMS	PEN
LANGUAGES	PAD

Example-1

- Get the names of employees who work in the DBSYSTEMS department and earn more than 35000

T1=SL EMPLOYEE
DEPT="DBSYSTEMS" AND SALARY>35000

T2=PJ T1
ENAME

Example - 2.1

- Get the names of suppliers who supply pads
- With selection

T1=PJ SUPPLY
SUPPLIER, ITEM

T2=SL T1
ITEM='PAD'

T3=PJ T2
SUPPLIER

T4=SUPPLIER JN T3
SNO=SUPPLIER

T5=PJ T4
SNAME

Other Operations in Relational Algebra

- Insertion
  - Union of the relation with the tuple
- Deletion
  - Difference of the relation from that tuple
- Modification
  - Insertion followed deletion

SQL

- Block – Structured
- English Language Keywords

Single Block Features

- Typically a trio of “SELECT-FROM-WHERE” keywords
- Syntax

```
SELECT [DISTINCT/UNIQE] fields(s)
FROM relation(s)
[WHERE predicate]
[GROUP BY field(s) [HAVING predicate]]
[ORDER BY field(s)]
```

SELECT Statement -1

- SELECT *

  - in single-block

    - an entire tuple to be output

  - in nested-block

    - following EXISTS keyword means the output of only the primary key of a tuple

SELECT Statement - 2

- SELECT DISTINCT SALARY

  - attribute name, when proceded with DISTINCT or UNIQUE, implies projection PJ relation

- SELECT DNAME, FLOOR

  - list of attributes will be the output

- SELECT COUNT(*)

    - outputs the number tuples(cardinality)

SELECT Statement - 3

- SELECT COUNT(DISTINCT DNAME)
  - outputs the distinct number of DNAME values
- SELECT SALARY*9/5-32
  - arithmetic value expression
- SELECT with aggregate functions
  - sum, max, min, average

FROM Statement

- Complements SELECT statement by specifying which relations the scope of the SELECT
- SELECT-FROM clause performs vertical subsetting and projection when DISTINCT keyword is used

WHERE Clause - 1

- Performs horizontal subsetting
- Reduces the input relation(s) to result relation by using a predicate expression
- Predicates

  - a simple predicate

    * N Θ opr where Θ is one of =, ≠, ≤,<, ≥,>

    * N Θ SQL where SQL is a single-block(in nested-block features)

Example-1

```
SELECT  DEPT, MGR
FROM EMPLOYEE
```

- Only vertical subsetting

Example-2

```
SELECT DISTINCT DEPT, MGR
FROM EMPLOYEE
```

- Vertical subsetting with projection

Example-3

```
SELECT DEPT,MGR
FROM EMPLOYEE
WHERE SALARY>40000
```

- Horizontal subsetting with a predicate of SALARY>40000

Example-4

```
SELECT DISTINCT MGR, DEPT
FROM EMPLOYEE
WHERE SALARY≤40000 AND
                        DEPT=“DBSYSTEMS”

(OR DEPT=“STORAGESYS”)
```

Example-5

```
SELECT ENAME
FROM EMPLOYEE
WHERE SALARY IS IN
                    (40000,39000,75000)
```

Example-6

```
SELECT COUNT(*)
FROM EMPLOYEE
WHERE SALARY>AVERAGE(SALARY)
```

- Use of aggregate functions

Example-7

```
SELECT DEPT
FROM EMPLOYEE
WHERE DEPT ≠ “STORAGESYS”
GROUP BY DEPT
HAVING COUNT(*) ≥ 3
```

- Use of GROUP BY … HAVING clauses

Example-8

```
SELECT E1.ENAME
FROM EMPLOYEE E1, EMPLOYEE E2
WHERE E1.SALARY>E2.SALARY
AND E2.ENAME="J.BRINK"
ORDER BY ENAME
```

- Use of aliases(tuple variables) and ORDER BY clauses
- By default ORDER BY is ascending

Example-9

```
SELECT ENAME
FROM EMPLOYEE, DEPT
WHERE  EMPLOYEE.DEPT=DEPT.DNAME
AND DEPT.FLOOR=2
```