Relational Algebra and SQL
2.4 and 6.1.

Recall:

Relational Algebra (RA)

- Operations on Relations.

Projection

$\pi_{\langle \text{List Expr} \rangle} R$

list of expressions on the attributes of a relation.

Ex. $R(a,b)$

a	b
1	9
3	3

① $\pi_a R$

a
1
3

② $\pi_{a+5,\,-b} R$

name of attributes

"a+5"	"-b"
6	-9
8	-3

(3) $\Pi_{b,a} R$

b	a
9	1
3	3

(4) $\Pi_{-1,a} R$

"-1"	a
-1	1
-1	3

SQL:

select <list expr> from R

(1) SELECT a FROM R

(2) SELECT a+5, -b FROM R

(3) SELECT b, a FROM R

(4) SELECT -1, a FROM R

Name of Relation optional!!

SELECT 3;

"3"
3

Creates table of one tuple!!

SELECT "abc", 5.2

=>

"abc"	"5.2"
'abc'	5.2

name of attributes.

Tuples.

The result of SELECT is always a relation

Renaming Relations and their attributes.

Sometimes we need to rename tables or their attributes.

$$\rho_{\langle new\ schema \rangle} R$$

Ex: $R(a,b)$ $\rho_{S(c,d)} R$ renames $R(a,b)$ to $S(c,d)$

dmg notation: you can rename during the projection.

If we want to rename the projected expression we can do it:

$$(\Pi_{a \to c,\ b \to d})\ R \to S$$

Result schema $S(c,d)$

Ex:

1 $\Pi_{a+5 \to x, -b \to y} R.$

x	y
6	-9
8	-3

SQL.

Given $R(a,b)$

$(\Pi_{a \to c, b \to d} R) \to S$

$\rho_{S(c,d)} R$

SELECT a, b FROM R as S(c,d)

or

SELECT a AS c, b as d FROM R

1

SELECT a + 5 AS X, -b AS Y FROM R

SELECTION

$$\sigma_p \; R$$

p is a predicate on attributes of R

Expressions:

$<$, $>$, $<>$, $=$, $>=$, $<=$

different equal

AND, NOT and many others.

Ex:

R(a,b)

a	b
3	2
1	∅

p evaluated at each tuple.

1 $\sigma_{a>1 \text{ OR } b>1} \; R$

a	b
3	2

SQL.

SELECT * FROM R WHERE p

original attributes of R

Ex:

1 SELECT * FROM R
WHERE a > 1 OR b > 1

We can combine Π and σ:

Ex: $\Pi_a \, \sigma_{a>1 \text{ OR } b>1} R$.

SELECT a FROM R
WHERE a>1 OR b>1

NOT equivalent to.

$\sigma_{a>1 \text{ OR } \underline{b>1}} \Pi_a R$.

b is not part of $\Pi_a R$.

$\boxed{\Pi \text{ and } \sigma \text{ are NOT distributive}}$

Questions

What does this return?

1) $\sigma_{FALSE} R$

2) $\sigma_{TRUE} R$

Other expressions in predicates.

IN

att IN (List)

Ex..

a IN (3, 2, 5)

$\Rightarrow$ equivalent to (a = 3 or a = 2 ~~or~~ a = 5)

But we can also use a query:

a in (π_c S)

SQL:

a IN (SELECT c FROM S)

EXISTS

EXISTS (R) true if R not empty

Ex:

EXISTS (σ_{ass} R)

Would return true if $|\sigma_{ass} R| > 0$

$|R|$ Represents # of tuples in relation R.

Operations on 2 Relations.

Union	$\cup$
Intersection	$\cap$
Difference (Except)	$-$

Union Compatible

R and S are "union compatible" iff

$$|attrs(R)| = |attrs(S)|$$

and the type of the i-th attribute of S. is type compatible with the type of the i-th attribute of R.

One type t_1 is type compatible with type t_2 if t_1 can be converted to type t_2.

$$\left.\begin{array}{l} A \cup B \\ A \cap B \\ A - B \end{array}\right\} \text{Defined only iff}$$

A & B are union compatible.

UNION

$t \in R \cup S \iff t \in R$ and $t \in S$

$t \in R \cap S \iff t \in R$ or $t \in S$

$t \in R - S \iff t \in R$ and $t \notin S$

Schema of result is schema of first relation.

Ex:

$R(a,b)$

a	b
1	a
3	x

$S(c,d)$

c	d
1	e
3	x
4	f

$R \cup S$

a	b
1	a
3	x
1	e
4	f

$R \cap S$

a	b
3	x

$R - S$

a	b
1	a

$S - R$

c	d
1	e
4	f

SQL

$$\text{TABLE R} \left\{ \begin{array}{l} \text{UNION} \\ \text{INTERSECT} \\ \text{EXCEPT} \end{array} \right\} \text{TABLE S}$$

or

$$\text{SELECT * FROM R} \left\{ \begin{array}{l} \text{UNION} \\ \text{INTERSECT} \\ \text{EXCEPT} \end{array} \right\}$$

$$\text{SELECT * FROM S}$$