

Modifying content

- Newly created relation is empty. Insert by

```
insert into employee values (Smith, 2000.00, 2, 3)
```

- Delete all tuples

```
delete from employee
```

- Remove a relation completely

```
drop table r
```

- Alter the schema (add attributes)

```
alter table r add A D
```

```
alter table r drop A
```

```
alter table employee add salary numeric(8, 2)
```

```
alter table employee drop salary
```

SQL Queries

- **select, from, where:** the 3 clauses.

- select = *project* of algebra. Lists the attributes for the query result
- from = *cartesian* product. Relations to be used for processing query
- where = *selection* predicate

- Notice the difference in meanings of **select** in algebra vs. SQL. The select of SQL is the project of algebra, and the select of algebra is the *where* of SQL.

Queries are of the form:

```
select A1, A2, ..., An  
from r1, r2, ..., rm  
where P
```

A_i 's are attributes, r_i 's are relations, and P is the predicate (condition). Therefore, equivalent to:

$$\Pi_{A_1, A_2, \dots, A_n}(\sigma_P(r_1 \times r_2 \times \dots \times r_m))$$

If **where** is omitted, P is true.

select clause

Query: Find the names of all departments:

```
select dept_name
from department
```

Note: Rel. algebra: mathematic $\implies$ really strict about sets. Therefore, **no duplicate values** (tuples) appear as a result of an expression/operation. *Reality:* duplicate elimination is time-wasting. So, SQL etc. **allow dupe tuples in relations** (but of course not for a col. which is primary key) and for **result of expressions**.

So, there may be dupe `dept_name` entries.

Otherwise:

```
select distinct dept_name
from department
```

Redundant:

```
select all dept_name
from department
```

“*” $\implies$ all attrbts.

```
select *
from department
```

or

```
select department.*
from department
```

select may include arthmtc: +, -, /, *

```
select name, salary * 1.1
from employee
```

where clause

Gives the predicate.

Query: Employee name with salary > 2000 and work in department with id=3

```
select name
from employee
where salary > 2000 and dept_id = 3
```

Also, **between** comparison operator:

```
select name
from employee
where salary between 2000 and 3000
```

Also, not between.

from clause

Does **CProduct** of all relations in clause.

Since Join = CP + select + project $\implies$ easy to do this in SQL.

For e.g., “Find the names of all employees who are in a team and their team names.”

```
select name, team_name
from employee, team
where employee.team_id = team.team_id
```

More complicated Q: “Find the names of all employees and their team names who are in a team that has at least three wins.”

```
select name, team_name
from employee, team
where employee.team_id = team.team_id and team.wins > 3
```

Rename operation

- Can rename **relations** and **attributes**. $\implies$ appears in both **select** and **from**.
- *old_name as new_name*
- Cases in which can't derive attrbt. names w/ **from** clause easily:
 - 2 relations in **from** w/ same attrbt. name
 - w/ arithmetic operatns

- may just wanna change names

Using *rename* in **select**:

```
select name, team_name as strong_team
from employee, team
where employee.team_id = team.team_id and team.wins > 3
```

Tuple variables

- Get TV's by renaming relations in **from**.
- Common in comparing attrs. in same relation:
 - e.g. Query: Name of teams who have more wins than at least one other:

```
select T.team_name
from team as T, team as S
where T.wins > S.wins
```

String operations

- Single quotes for strings: “ ‘ ”; **Case sensitive**
- **like** for **pattern matching**
 - % : matches any substring
 - _ : matches any character
 - * ‘Day%’ matches ‘Dayton’, ‘Days’, ‘Day’ ‘Day 12’
 - * ‘%Day%’ matches ‘12th Day’, ‘Today’, ‘Day’
 - * ‘%Day_’ matches ‘24 Days’
 - * ‘___’ matches a string with exactly 3 chars.
 - * ___% matches a string with at least 3 chars.

```
select name
from employee
where name like '%Main%'
```

- Use \ to escape % and _ in strings
- **not like** operator to search for **mismatches**.