

Carnegie Mellon Univ.
Dept. of Computer Science
15-415 - Database Applications

C. Faloutsos
Rel. model - SQL part1

Carnegie Mellon

General Overview - rel. model

- Formal query languages
 - rel algebra and calculi
- Commercial query languages
 - SQL
 - QBE, (QUEL)

Carnegie Mellon

15-415 - C. Faloutsos

2

Overview - detailed - SQL

- DML
 - select, from, where, renaming
 - set operations
 - ordering
 - aggregate functions
 - nested subqueries
- other parts: DDL, embedded SQL, auth etc

DML

General form

```
select a1, a2, ... an  
from r1, r2, ... rm  
where P  
[order by ...]  
[group by ...]  
[having ...]
```

Reminder: our Mini-U db

STUDENT		
<u>Ssn</u>	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
<u>c-id</u>	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
<u>SSN</u>	<u>c-id</u>	grade
123	15-413	A
234	15-413	B

DML - eg:

find the ssn(s) of everybody called “smith”

select ssn

from student

where name=“smith”

DML - observation

General form

select a_1, a_2, \dots, a_n
from r_1, r_2, \dots, r_m
where P

equivalent rel. algebra query?

DML - observation

General form

select a_1, a_2, \dots, a_n
from r_1, r_2, \dots, r_m
where P

$\pi_{a_1, a_2, \dots, a_n} (\sigma_P (r_1 \times r_2 \times \dots \times r_m))$

DML - observation

General form

select distinct a_1, a_2, \dots, a_n
from r_1, r_2, \dots, r_m
where P

$\pi_{a_1, a_2, \dots, a_n} (\sigma_P (r_1 \times r_2 \times \dots \times r_m))$

select clause

select [**distinct** | **all**] name
from student
where address="main"

where clause

find ssn(s) of all “smith”s on “main”

select ssn

from student

where address=“main” **and**

name = “smith”

where clause

- boolean operators (**and or not ...**)
- comparison operators (<, >, =, ...)
- and more...

What about strings?

find student ssns who live on “main” (st or str
or street - ie., “main st” or “main str” ...)

What about strings?

find student ssns who live on “main” (st or str
or street)

select ssn

from student

where address **like** “main%”

%: variable-length don't care

_: single-character don't care

from clause

find names of people taking 15-415

STUDENT			CLASS		
<u>Ssn</u>	Name	Address	<u>c-id</u>	c-name	units
123	smith	main str	15-413	s.e.	2
234	jones	forbes ave	15-412	o.s.	2

TAKES		
<u>SSN</u>	<u>c-id</u>	<u>grade</u>
123	15-413	A
234	15-413	B

from clause

find names of people taking 15-415

select name

from student, takes

where ???

from clause

find names of people taking 15-415

select name

from student, takes

where student.ssn = takes.ssn **and**
takes.c-id = "15-415"

renaming - tuple variables

find names of people taking 15-415

select name

from ourVeryOwnStudent, studentTakingClasses

where ourVeryOwnStudent.ssn =
studentTakingClasses.ssn

and studentTakingClasses.c-id = "15-415"

renaming - tuple variables


find names of people taking 15-415

```
select name
from ourVeryOwnStudent as S,
     studentTakingClasses as T
where S.ssn = T.ssn
     and T.c-id = "15-415"
```

renaming - self-join

- self -joins: find Tom's grandparent(s)

PC	
<u>p-id</u>	<u>c-id</u>
Mary	Tom
Peter	Mary
John	Tom



PC	
<u>p-id</u>	<u>c-id</u>
Mary	Tom
Peter	Mary
John	Tom

renaming - self-join

find grandparents of “Tom” (PC(p-id, c-id))

```
select gp.p-id  
from PC as gp, PC  
where gp.c-id= PC.p-id  
and PC.c-id = “Tom”
```

renaming - theta join

find course names with more units than 15-415

```
select c1.c-name  
from class as c1, class as c2  
where c1.units > c2.units  
and c2.c-id = “15-415”
```

renaming - theta join

find course names with more units than 15-415

```

select c1.c-name
from class as c1, class as c2
where c1.units > c2.units
and c2.c-id = "15-415"

```

find course names with more units than 15-415

```

select c1.name
from class as c1, class as c2
where c1.units > c2.units
and c2.c-id = "15-415"

```

$$\{t \mid \exists c1 \in CLASS \quad \exists c2 \in CLASS \quad ($$

$$c1[c - id] = 15 - 415 \wedge$$

$$c2[units] > c1[units] \wedge$$

$$t[c - name] = c2[c - name])\}$$

Overview - detailed - SQL

- DML
 - select, from, where
 - set operations
 - ordering
 - aggregate functions
 - nested subqueries
- other parts: DDL, embedded SQL, auth etc

set operations

find ssn of people taking both 15-415 and 15-413

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

set operations

find ssn of people taking both 15-415 and 15-413

```
select ssn  
from takes  
where c-id="15-415" and  
c-id="15-413"
```

set operations

find ssn of people taking both 15-415 and 15-413

```
(select ssn from takes where c-id="15-415" )
```

intersect

```
(select ssn from takes where c-id="15-413" )
```

other ops: **union** , **except**

Overview - detailed - SQL

- DML
 - select, from, where
 - set operations
 - ordering
 - aggregate functions
 - nested subqueries
- other parts: DDL, embedded SQL, auth etc

Ordering

find student records, sorted in name order

```
select *  
from student  
—where—
```

Ordering

find student records, sorted in name order

```
select *  
from student  
order by name asc
```

asc is the default

Ordering

find student records, sorted in name order;

break ties by reverse ssn

```
select *  
from student  
order by name, ssn desc
```


Overview - detailed - SQL

- DML
 - select, from, where
 - set operations
 - ordering
 - aggregate functions
 - nested subqueries
- other parts: DDL, embedded SQL, auth etc

Aggregate functions

find avg grade, across all students

select ??

from takes

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions

find avg grade, across all students

select avg(grade)

from takes

SSN	c-id	grade
123	15-413	4
234	15-413	3

- result: a single number
- Which other functions?

Aggregate functions

- A: **sum count min max (std)**

Aggregate functions

find total number of enrollments

select count(*)

from takes

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions

find total number of students in 15-415

select count(*)

from takes

where c-id="15-415"

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions

find total number of students in each course

select count(*)

from takes

where ???

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions

find total number of students in each course

select c-id, count(*)

from takes

group by c-id

SSN	c-id	grade
123	15-413	4
234	15-413	3

c-id	count
15-413	2

Aggregate functions

find total number of students in each course

select c-id, **count**(*)

from takes

group by c-id

order by c-id

SSN	c-id	grade
123	15-413	4
234	15-413	3

c-id	count
15-413	2

Aggregate functions

find total number of students in each course,
and sort by count, decreasing

select c-id, **count**(*) **as** pop

from takes

group by c-id

order by pop desc

SSN	c-id	grade
123	15-413	4
234	15-413	3

c-id	pop
15-413	2

Aggregate functions- 'having'

find students with GPA > 3.0

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions- 'having'

find students with GPA > 3.0

select ???, **avg**(grade)

from takes

group by ???

SSN	c-id	grade
123	15-413	4
234	15-413	3

Aggregate functions- 'having'

find students with GPA > 3.0

select ssn, avg(grade)

from takes

group by ssn

???

SSN	c-id	grade
123	15-413	4
234	15-413	3

SSN	avg(grade)
123	4
234	3

Aggregate functions- 'having'

find students with GPA > 3.0

select ssn, avg(grade)

from takes

group by ssn

having avg(grade)>3.0

SSN	c-id	grade
123	15-413	4
234	15-413	3

SSN	avg(grade)
123	4
234	3

'having' <-> 'where' for groups

Aggregate functions- 'having'

find students and GPA,
for students with > 5 courses

```
select ssn, avg(grade)
from takes
group by ssn
having count(*) > 5
```

SSN	c-id	grade
123	15-413	4
234	15-413	3

SSN	avg(grade)
123	4
234	3

Overview - detailed - SQL

- DML
 - select, from, where
 - set operations
 - ordering
 - aggregate functions
 - nested subqueries
- other parts: DDL, embedded SQL, auth etc