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

Fall 2010 (Kesden)

Slides from Christos Faloutsos

General Overview - rel. model

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

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		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

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

DML - nested subqueries

find names of students of 15-415

select name

from student

where ...

"ssn in the set of people that take 15-415"

DML - nested subqueries

find names of students of 15-415

select name

from student

where ……..

select ssn

from takes

where c-id =“15-415”

DML - nested subqueries

find names of students of 15-415

```
select name
from student
where  ssn in (
   select ssn
   from takes
   where  c-id =“15-415”)
```

DML - nested subqueries

- **'in'** compares a value with a set of values
- **'in'** can be combined other boolean ops
- it is redundant (but user friendly!):

select name

from student …..

where c-id = "15-415" ….

DML - nested subqueries

- **'in'** compares a value with a set of values
- **'in'** can be combined other boolean ops
- it is redundant (but user friendly!):

select name
from student, takes
where c-id = "15-415" **and**
student.ssn=takes.ssn

DML - nested subqueries

find names of students taking 15-415 and living on “main str”

select name

from student

where address=“main str” **and** ssn **in**

(**select** ssn **from** takes **where** c-id =“15-415”)

DML - nested subqueries

- **'in'** compares a value with a set of values
- other operators like '**in**' ??

DML - nested subqueries

find student record with highest ssn

select *

from student

where ssn

is greater than every other ssn

DML - nested subqueries

find student record with highest ssn

select *

from student

where ssn *greater than every*

select ssn **from** student

DML - nested subqueries

find student record with highest ssn

```
select *
from student
where ssn  > all (
    select ssn   from student)
```

almost correct

DML - nested subqueries

find student record with highest ssn

```
select *
from student
where ssn  >= all (
    select ssn   from student)
```

DML - nested subqueries

find student record with highest ssn - without nested subqueries?

select S1.ssn, S1.name, S1.address

from student **as** S1, student **as** S2

where S1.ssn > S2.ssn

is not the answer (what does it give?)

DML - nested subqueries

S1

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

S2

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

S1 x S2

S1. ssn	S2.ssn	….
123	123	…
234	123	…
123	234	
234	234	

S1.ssn>S2.ssn

DML - nested subqueries

select S1.ssn, S1.name, S1.address
from student **as** S1, student **as** S2
where S1.ssn > S2.ssn

gives all but the smallest ssn -

aha!

DML - nested subqueries

find student record with highest ssn - without nested subqueries?

select S1.ssn, S1.name, S1.address

from student **as** S1, student **as** S2

where S1.ssn < S2.ssn

gives all but the highest - therefore….

DML - nested subqueries

find student record with highest ssn - without nested subqueries?

```
(select * from student)      except
(select S1.ssn, S1.name, S1.address
 from student as S1, student as S2
 where S1.ssn < S2.ssn)
```

DML - nested subqueries

```
(select *  from student)      except
(select S1.ssn, S1.name, S1.address
 from student as S1, student as S2
 where S1.ssn < S2.ssn)
```

```
select *
from student
where ssn  >= all (select ssn   from student)
```

DML - nested subqueries

Drill: Even more readable than

select * **from** student

where ssn >= **all** (**select** ssn **from** student)

DML - nested subqueries

Drill: Even more readable than

select * **from** student
where ssn >= **all** (**select** ssn **from** student)

select * **from** student
where ssn **in**
(**select max**(ssn) **from** student)

DML - nested subqueries

Drill: find the ssn of the student with the highest GPA

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

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

DML - nested subqueries

Drill: find the ssn and GPA of the student with the highest GPA

select ssn, **avg**(grade) **from** takes

~~**where**~~

DML - nested subqueries

Drill: find the ssn and GPA of the student with the highest GPA

select ssn, **avg**(grade) **from** takes

group by ssn

having avg(grade**) ……**

greater than every other GPA on file

DML - nested subqueries

Drill: find the ssn and GPA of the student with the highest GPA

```
select ssn, avg(grade) from takes
group by ssn
having avg( grade) >=   all
    ( select avg( grade )
      from student group by ssn )
```

} all GPAs

DML - nested subqueries

- **'in'** and **'>= all'** compares a value with a set of values
- other operators like these?

DML - nested subqueries

- **<all(), <>all()** ...
- '**<>all**' is identical to '**not in**'
- **>some(), >= some** () ...
- '**= some()**' is identical to '**in**'
- **exists**

DML - nested subqueries

Drill for **'exists'**: find all courses that nobody enrolled in

select c-id **from** class ….*with no tuples in 'takes'*

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

CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

DML - nested subqueries

Drill for '**exists**': find all courses that nobody enrolled in

```
select c-id from class
where not exists
    (select * from  takes
    where class.c-id = takes.c-id)
```

DML - derived relations

find the ssn with the highest GPA

```
select ssn, avg(grade) from takes
group by ssn
having avg( grade) >=   all
     ( select avg( grade )
        from takes group by ssn )
```

DML - derived relations

find the ssn with the highest GPA

Query would be easier, if we had a table like:
helpfulTable (ssn, gpa):

HelpfulTable	
Ssn	Gpa
123	3.5
678	3.3

then what?

DML - derived relations

HelpfulTable	
Ssn	Gpa
123	3.5
678	3.3

select ssn, gpa
from helpfulTable
where gpa **in** (**select** **max**(gpa)
from helpfulTable)

DML - derived relations

find the ssn with the highest GPA -

Query for helpfulTable (ssn, gpa)?

DML - derived relations

find the ssn with the highest GPA

Query for helpfulTable (ssn, gpa)?

select ssn, **avg**(grade)
from takes
group by ssn

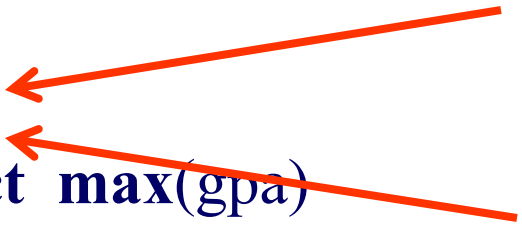
DML - derived relations

find the ssn with the highest GPA

helpfulTable(ssn,gpa)

```
select ssn, gpa
from helpfulTable
where gpa = (select  max(gpa)
             from helpfulTable)
```

```
select ssn, avg(grade)
from takes
group by ssn
```



DML - derived relations

find the ssn with the highest GPA

```
select ssn, gpa
from (select ssn, avg(grade)
        from takes
        group by ssn)
        as helpfulTable(ssn, gpa)
where gpa in (select  max(gpa)
                from helpfulTable)
```

Views

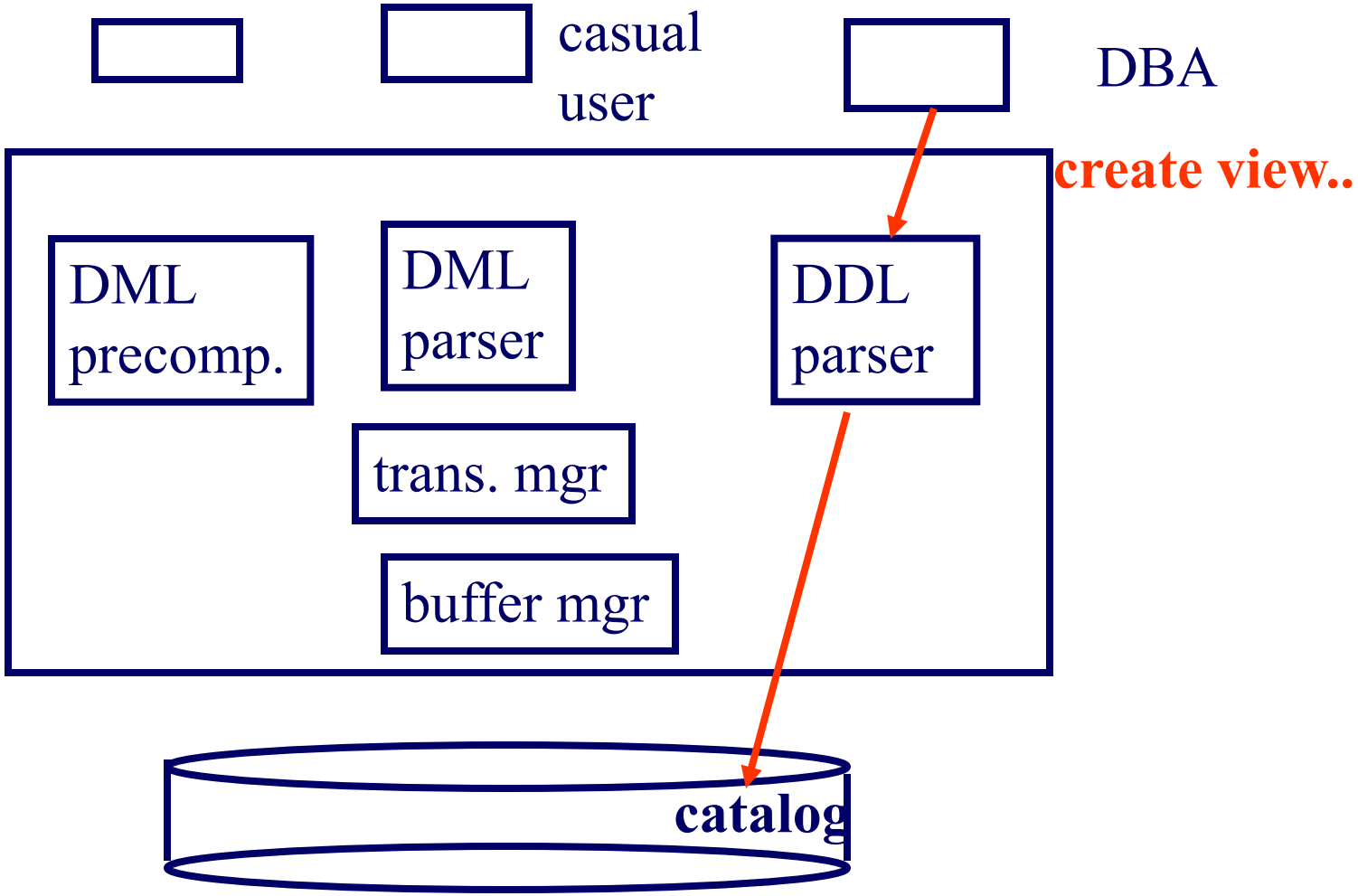
find the ssn with the highest GPA -
we can create a permanent, virtual table:

```
create view helpfulTable(ssn, gpa)  as
        select ssn, avg(grade)
         from takes
         group by ssn
```

Views

- views are recorded in the schema, for ever (ie., until '**drop view**…')
- typically, they take little disk space, because they are computed on the fly
- (but: materialized views…)

Overview of a DBMS



Overview - detailed - SQL

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

Overview - detailed - SQL

- DML
- other parts:
  - modifications
  - joins
  - DDL
  - embedded SQL
  - authorization