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

C. Faloutsos
Rel. model - SQL part 2

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

<table>
<thead>
<tr>
<th>STUDENT</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>id</td>
</tr>
<tr>
<td>123</td>
<td>smith</td>
</tr>
<tr>
<td>234</td>
<td>jones</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TAKES</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>234</td>
</tr>
</tbody>
</table>

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

```sql
select name
from student
where ..........
select ssn
from takes
where c-id = '15-415'
```

DML - nested subqueries

find names of students of 15-415

```sql
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!):
  ```sql
  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!):
  ```sql
  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"

```sql
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  
    )
```

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

\[
\text{select } S1\text{.ssn, } S1\text{.name, } S1\text{.address}
\text{from student as } S1, \text{ student as } S2
\text{where } S1\text{.ssn} > S2\text{.ssn}
\]
gives all but the smallest ssn - aha!

DML - nested subqueries

\[
\text{find student record with highest ssn - without nested subqueries?}
\text{(select } *) \text{ from student) except}
\text{(select } S1\text{.ssn, } S1\text{.name, } S1\text{.address}
\text{from student as } S1, \text{ student as } S2
\text{where } S1\text{.ssn} < S2\text{.ssn})
\]

DML - nested subqueries

Drill: Even more readable than
\[
\text{select } * \text{ from student}
\text{where ssn } \geq \text{ all (select ssn from student)}
\]

DML - nested subqueries

\[
\text{find student record with highest ssn - without nested subqueries?}
\text{select } S1\text{.ssn, } S1\text{.name, } S1\text{.address}
\text{from student as } S1, \text{ student as } S2
\text{where } S1\text{.ssn} < S2\text{.ssn}
\]
gives all but the highest - therefore...

DML - nested subqueries

\[
\text{select } * \text{ from student) except}
\text{(select } S1\text{.ssn, } S1\text{.name, } S1\text{.address}
\text{from student as } S1, \text{ student as } S2
\text{where } S1\text{.ssn} < S2\text{.ssn})
\]

select *
\text{from student}
\text{where ssn } \geq \text{ all (select ssn from student)}

DML - nested subqueries

Drill: Even more readable than
\[
\text{select } * \text{ from student}
\text{where ssn } \geq \text{ all (select ssn from student)}
\]

\[
\text{select } * \text{ from student}
\text{where ssn in}
\text{(select max(ssn) from student)}
\]
DML - nested subqueries

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

<table>
<thead>
<tr>
<th>STUDENT</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ssn</td>
<td>Name</td>
</tr>
<tr>
<td>123</td>
<td>Smith</td>
</tr>
<tr>
<td>234</td>
<td>Jones</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TAKES</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>234</td>
</tr>
</tbody>
</table>

DML - nested subqueries

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

\[
\text{select ssn, } \text{avg(grade) from takes}
\]

\[
\text{where}
\]

DML - nested subqueries

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

\[
\text{select ssn, } \text{avg(grade) from takes}
\]

\[
\text{group by ssn}
\]

\[
\text{having avg(grade) } \geq \text{ all}
\]

\[
(\text{select avg(grade) from student group by ssn}) \}\text{all GPAs}
\]

DML - nested subqueries

- "in" and "\geq all" compares a value with a set of values
- other operators like these?
**DML - nested subqueries**

Drill for "exists": find all courses that nobody enrolled in

```sql
select c-id from class ...with no tuples in "takes"
```

<table>
<thead>
<tr>
<th>TAKES</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>c-id</td>
</tr>
<tr>
<td>123</td>
<td>15-413 A</td>
</tr>
<tr>
<td>234</td>
<td>15-413 B</td>
</tr>
<tr>
<td>15-413</td>
<td>S.E.</td>
</tr>
<tr>
<td>15-412</td>
<td>O.S.</td>
</tr>
</tbody>
</table>

**DML - derived relations**

find the ssn with the highest GPA

```sql
select ssn, avg(grade) from takes
having avg(grade) >= all
(select avg(grade) from student 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):
```

<table>
<thead>
<tr>
<th>HelpfulTable</th>
<th>Ssn</th>
<th>Gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ssn</td>
<td>Gpa</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>678</td>
<td>3.3</td>
<td></td>
</tr>
</tbody>
</table>

then what?

**DML - derived relations**

```sql
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)?

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

Views

- views are recorded in the schema, forever (i.e., until 'drop view...')
- typically, they take little disk space, because they are computed on the fly
- (but: materialized views...)

Overview of a DBMS

Casual user

DML parser

DBA

create view...

DML parser

buffer mgr

DBA

index mgr
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