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

\[
\text{select } a_1, a_2, \ldots, a_n \\
\text{from } r_1, r_2, \ldots, r_m \\
\text{where } \text{P} \\
\text{[order by …]} \\
\text{[group by …]} \\
\text{[having …]}
\]

Reminder: our Mini-U db

<table>
<thead>
<tr>
<th>STUDENT</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ssn</td>
<td>c-id</td>
</tr>
<tr>
<td>123 smith</td>
<td>15-413</td>
</tr>
<tr>
<td>234 jones</td>
<td>15-412</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TAKES</th>
<th>SSN</th>
<th>c-id</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>123 15-413</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>234/15-413</td>
<td>B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DML - nested subqueries

find names of students of 15-415

\[
\text{select name} \\
\text{from student} \\
\text{where } \ldots
\]

“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
```

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

\[
\text{select * } \\
\text{from student} \\
\text{where ssn > all (select ssn from student)}
\]

almost correct
DML - nested subqueries

find student record with highest ssn

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

DML - nested subqueries

find student record with highest ssn - without nested subqueries?

```sql
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?)

<table>
<thead>
<tr>
<th>S1</th>
<th>S2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STUDENT</strong></td>
<td><strong>STUDENT</strong></td>
</tr>
<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>S1 x S2</th>
<th>S1.ssn</th>
<th>S2.ssn</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>123</td>
<td>...</td>
</tr>
<tr>
<td>234</td>
<td>123</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S1.ssn&gt;S2.ssn</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
</tr>
<tr>
<td>234</td>
</tr>
</tbody>
</table>
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

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

\[
\begin{align*}
&\text{select * from student where ssn >= all (select ssn from student)}
\end{align*}
\]

DML - nested subqueries

Drill: Even more readable than

\[
\begin{align*}
&\text{select * from student} \\
&\text{where ssn >= all (select ssn from student)}
\end{align*}
\]

select * from student

where ssn in

(select max(ssn) from student)

DML - nested subqueries

Drill: Even more readable than

\[
\begin{align*}
&\text{select * from student} \\
&\text{where ssn >= all (select ssn from student)}
\end{align*}
\]
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>c-id</td>
</tr>
<tr>
<td>123</td>
<td>16-413</td>
</tr>
<tr>
<td>234</td>
<td>16-412</td>
</tr>
<tr>
<td>smith</td>
<td>main str</td>
</tr>
<tr>
<td>jones</td>
<td>forbes ave</td>
</tr>
</tbody>
</table>

DML - nested subqueries

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

```sql
select ssn, avg(grade) from takes
where greater than every other GPA on file
```

DML - nested subqueries

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

```sql
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

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

DML - nested subqueries

• ‘in’ and ‘\( \geq \text{all} \)’ 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

\[
\text{select } c\text{-id from class \ldots with no tuples in ‘takes’}
\]

<table>
<thead>
<tr>
<th>SSN</th>
<th>c-id</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>15-413</td>
<td>A</td>
</tr>
<tr>
<td>234</td>
<td>15-413</td>
<td>B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C-ID</th>
<th>C-NAME</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-413</td>
<td>S.E.</td>
<td>2</td>
</tr>
<tr>
<td>15-412</td>
<td>O.S.</td>
<td>2</td>
</tr>
</tbody>
</table>

DML - nested subqueries

Drill for ‘exists’: find all courses that nobody enrolled in

\[
\begin{align*}
\text{select} & \quad c\text{-id from class} \\
\text{where} & \quad \text{not exists} \\
& \quad \left( \text{select } * \text{ from takes} \\
& \quad \quad \text{where class.c\text{-id} = takes.c\text{-id}} \right)
\end{align*}
\]

DML - derived relations

find the ssn with the highest GPA

\[
\begin{align*}
\text{select} & \quad \text{ssn, avg(grade) from takes} \\
\text{group by} & \quad \text{ssn} \\
\text{having} & \quad \text{avg(grade)} \geq \text{all} \\
& \quad \left( \text{select avg(grade) from takes group by ssn} \right)
\end{align*}
\]
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>Ssn</th>
<th>Gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>3.5</td>
</tr>
<tr>
<td>678</td>
<td>3.3</td>
</tr>
</tbody>
</table>

then what?

DML - derived relations
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

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

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:

```sql
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