Loading our example tables

- Look at `createdb.sql` (converted to HTML by <pre>)
- Note portable datatypes
- Note added “not null” constraints, missing in text.
- Same `createdb.sql` and `dropdb.sql` work for both Oracle and `mysql`.
- Table loading is different in the two cases
  - We could use inserts portably, but would run slower.
  - So we are using “bulk loads”, with special commands: `sqlldr` for Oracle and the load data command in `mysql`.

Createdb.sql

- Note that the tables form five groups, or “databases”:
  - Student, faculty, class, enrolled
  - Emp, dept, works
  - Flights, aircraft, employees, certified
  - Suppliers, parts, catalog
  - Sailors, boats, reserves
- In all 5 groups, the last-named table has two FKs to PKs of other tables: this is an indication of a relationship table:
  - A student is enrolled in a class
  - An emp works in a dept
  - An employee is certified to work on an aircraft
  - A part is cataloged by a supplier
  - A sailor reserves a boat

Nested Queries (subqueries)

- An SQL query can be used to help the evaluation of another query
  - E.g., a condition may need to be evaluated on a computed relation, not one readily available
  - Multiple levels of nesting are possible
  - Semantics similar to those of nested loops
- Nested queries do not appear in relational algebra
  - But it is possible to write relational algebra expressions to obtain same result (unless count, sum, etc. are in use)
  - Using nested queries leads to more concise solutions

Connecting queries and subqueries

- Subquery: Select … from … inside another Select … from …
- A subquery can return:
  - A scalar value (1x1 table) – such a subquery can appear in a query in the same place where a constant appears, in SQL-99-compliant DBs.
  - A relation
- Where can subqueries appear?
  - Most often in WHERE clause of parent query
  - Also used in FROM clause followed by range variable
  - … FROM Sailors, (SELECT bid FROM Boats) Bids …
  - In HAVING clauses
    - Will discuss later on
  - Also, if scalar, in the select-list. Not covered in text, and not in Entry-level SQL-92.

Example Schema

<table>
<thead>
<tr>
<th>Sailors</th>
<th>Boats</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>name</td>
</tr>
<tr>
<td>22</td>
<td>dustin</td>
</tr>
<tr>
<td>31</td>
<td>lubber</td>
</tr>
<tr>
<td>58</td>
<td>rusty</td>
</tr>
<tr>
<td>101</td>
<td></td>
</tr>
<tr>
<td>103</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserves</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
</tr>
<tr>
<td>22</td>
</tr>
<tr>
<td>58</td>
</tr>
</tbody>
</table>
Subqueries that return a constant

- Also referred to as subqueries that return a scalar
- Most easy case to understand

```
SELECT S.sname
FROM Sailors S 
WHERE S.sid = (SELECT R.sid
FROM Reserves R
WHERE R.bid=103)
```

- If subquery returns more than one value or zero values, a runtime error occurs! FRAGILE, AVOID! (OK if "select max(...) ...", etc.)

- Next, we focus on subqueries that return relations

Correlated subquery

- Test that a relation satisfies some condition

```
SELECT S.sname
FROM Sailors S 
WHERE EXISTS (SELECT * 
FROM Reserves R
WHERE R.bid=103 AND S.sid=R.sid)
```

- Find names of sailors who’ve reserved boat #103

- Subquery is CORRELATED with parent query

Conditions involving relations and tuples

- Typically have some sort of set operations semantics

```
WHERE field IN (SELECT … )
WHERE field op ANY (SELECT …) (alternative: SOME)
WHERE field op ALL (SELECT … )
```

```
SELECT S.sname
FROM Sailors S 
WHERE S.sid IN (SELECT R.sid
FROM Reserves R
WHERE R.bid=103)
```

- Find names of sailors who’ve reserved boat #103

How the ALL keyword works

<table>
<thead>
<tr>
<th>Condition</th>
<th>Equivalent expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>x &gt; ALL (1, 2)</td>
<td>x &gt; 2</td>
</tr>
<tr>
<td>x &lt; ALL (1, 2)</td>
<td>x &lt; 1</td>
</tr>
<tr>
<td>x = ALL (1, 2)</td>
<td>(x = 1) AND (x = 2)</td>
</tr>
<tr>
<td>x &lt;&gt; ALL (1, 2)</td>
<td>(x &lt;&gt; 1) AND (x &lt;&gt; 2)</td>
</tr>
</tbody>
</table>

How the ANY and SOME keywords work

<table>
<thead>
<tr>
<th>Condition</th>
<th>Equivalent expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>x &gt; ANY (1, 2)</td>
<td>x &gt; 1</td>
</tr>
<tr>
<td>x &lt; ANY (1, 2)</td>
<td>x &lt; 2</td>
</tr>
<tr>
<td>x = ANY (1, 2)</td>
<td>(x = 1) OR (x = 2)</td>
</tr>
<tr>
<td>x &lt;&gt; ANY (1, 2)</td>
<td>(x &lt;&gt; 1) OR (x &lt;&gt; 2)</td>
</tr>
</tbody>
</table>

Conditions involving relations and tuples

- Typically have some sort of set operations semantics

```
WHERE field IN (SELECT … )
WHERE field op ANY (SELECT …)
WHERE field op ALL (SELECT … )
```

```
SELECT S.sname
FROM Sailors S 
WHERE S.rating > ANY (SELECT S1.rating 
FROM Sailors S1, Reserves R1
WHERE S1.sid=R1.sid AND R1.bid=103)
```

- Find names of sailors whose rating is higher than the minimum rating among sailors who reserved boat 103
Conditions involving relations and tuples

- Typically have some sort of set operations semantics
  
  \[
  \text{...WHERE field IN (SELECT \ldots)}
  \]
  
  \[
  \text{...WHERE field op ANY (SELECT \ldots)}
  \]
  
  \[
  \text{...WHERE field op ALL (SELECT \ldots)}
  \]

- Find names of sailors with maximum age

Subqueries in the FROM clause

\[
\text{SELECT SQ.sname, SQ.bname}
\]

\[
\text{FROM ( SELECT S.sname, B.name AS bname }
\]

\[
\text{FROM Sailors S, Boats B, Reserves R}
\]

\[
\text{WHERE S.sid=R.sid and B.bid=R.bid ) SQ}
\]

\[
\text{WHERE SQ.bname='interlake'}
\]

- Find names of sailors who reserved 'interlake'

Rewriting INTERSECT Queries Using IN

Find sid’s of sailors who’ve reserved both a red and a green boat:

\[
\text{SELECT S.sid}
\]

\[
\text{FROM Sailors S, Boats B, Reserves R}
\]

\[
\text{WHERE S.sid=R.sid and R.bid=B.bid AND B.color='red'}
\]

\[
\text{AND S.sid IN (SELECT S2.sid}
\]

\[
\text{FROM Sailors S2, Boats B2, Reserves R2}
\]

\[
\text{WHERE S2.sid=R2.sid and R2.bid=B2.bid AND B2.color='green')}
\]

- Similarly, EXCEPTION queries re-written using NOT IN.

Example Query 1

<table>
<thead>
<tr>
<th>Sailors</th>
<th>Boats</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>sname</td>
</tr>
<tr>
<td>bid</td>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserves</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
</tr>
</tbody>
</table>

- Find sid’s of sailors who’ve reserved only red boats

Example Query 1 Answer

<table>
<thead>
<tr>
<th>Sailors</th>
<th>Boats</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>sname</td>
</tr>
<tr>
<td>bid</td>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserves</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
</tr>
</tbody>
</table>

- Find sid’s of sailors who’ve reserved only red boats

\[
\text{SELECT R.sid}
\]

\[
\text{FROM Reserves R, Boats B}
\]

\[
\text{WHERE B.bid = R.bid AND B.color='red' AND R.sid NOT IN (SELECT R1.sid FROM Reserves R1, Boats B1}
\]

\[
\text{WHERE B1.color<>‘red’ AND B1.bid = R1.bid)}
\]
Example Query 1 Answer Alternative

- **Sailors**
  - sid | sname | rating | age

- **Boats**
  - bid | name | color

- **Reserves**
  - sid | bid | day

Find sids of sailors who've reserved only red boats

```
SELECT R.sid FROM Reserves R
WHERE NOT EXISTS(
  SELECT * from Reserves R1, Boats B
  WHERE B.bid = R1.bid AND R1.sid = R.sid
  AND B.color <> 'red'
)
```

Example Query 1 Answer Alternative

- **Sailors**
  - sid | sname | rating | age

- **Boats**
  - bid | name | color

- **Reserves**
  - sid | bid | day

Find sids of sailors who've reserved only red boats

```
SELECT S.sid FROM Sailors S
WHERE NOT EXISTS(
  SELECT * from Reserves R1, Boats B
  WHERE B.bid = R1.bid AND R1.sid = S.sid
  AND B.color <> 'red'
)
```

This returns sailors with no reservations at all! (as well as sailors reserving only red boats)

Example Query 2

- **Sailors**
  - sid | sname | rating | age

- **Boats**
  - bid | name | color

- **Reserves**
  - sid | bid | day

Find the name(s) of sailor(s) who have the highest rating (without using MAX because we haven't covered it yet)

Recall trick from RA: self-join (cross product) with age1 < age2 to find age1 values under the max

Example Query 2 Answer

- **Sailors**
  - sid | sname | rating | age

- **Boats**
  - bid | name | color

- **Reserves**
  - sid | bid | day

Find the name(s) of sailor(s) who have the highest rating

```
SELECT S1.sname FROM Sailors S1
WHERE S1.sid NOT IN (
  SELECT S2.sid FROM Sailors S2, Sailors S3
  WHERE S2.rating < S3.rating
)
```

Example Query 2 Answer Alternatives

- **Sailors**
  - sid | sname | rating | age

- **Boats**
  - bid | name | color

- **Reserves**
  - sid | bid | day

Find the name(s) of sailor(s) who have the highest rating

```
SELECT S.sname FROM Sailors S
WHERE S.rating >= ALL (SELECT rating FROM Sailors)
```

Using max, not yet officially covered:

```
SELECT S.sname FROM Sailors S
WHERE S.rating = (SELECT max(rating) FROM Sailors)
```

A query that uses ALL

```
SELECT vendor_name, invoice_number, invoice_total
FROM invoices i JOIN vendors v
ON i.vendor_id = v.vendor_id
WHERE invoice_total > ALL
  (SELECT invoice_total
   FROM invoices
   WHERE vendor_id = 34)
ORDER BY vendor_name
```

The result set

<table>
<thead>
<tr>
<th>vendor_name</th>
<th>invoice_number</th>
<th>invoice_total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>134.94</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>208.38</td>
<td></td>
</tr>
</tbody>
</table>

(25 rows)
A query that uses ANY

```
SELECT vendor_name, invoice_number, invoice_total
FROM vendors JOIN invoices
ON vendors.vendor_id = invoices.invoice_id
WHERE invoice_total < ANY
(SELECT invoice_total FROM invoices
WHERE vendor_id = 115)
```

The result of the subquery

<table>
<thead>
<tr>
<th>invoice_total</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.45</td>
</tr>
<tr>
<td>15.45</td>
</tr>
<tr>
<td>15.45</td>
</tr>
</tbody>
</table>

The result set

<table>
<thead>
<tr>
<th>vendor_name</th>
<th>invoice_number</th>
<th>invoice_total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Office Plaza</td>
<td>3000311235</td>
<td>20.97</td>
</tr>
<tr>
<td>screams Haunted by the Bank 20032317</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Institute Communications Inc</td>
<td>24605756</td>
<td>0</td>
</tr>
<tr>
<td>Stomach Swirling Company</td>
<td>47858322</td>
<td>15.45</td>
</tr>
<tr>
<td>State Bears Revenue</td>
<td>112-434-10346</td>
<td>15.45</td>
</tr>
</tbody>
</table>