SQL Nested Queries

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 the same result.
- Using nested queries leads to more concise solutions.

Connecting queries and subqueries:
- Depends on what the subquery returns:
  - A scalar value (1x1 table) – can appear in a query in the same place where a constant appears.
  - A relation.
- Where can subqueries appear?
  - Most often in WHERE clause of parent query:
    - ... FROM Sailors, (SELECT bid FROM Boats) Bids ...
  - In HAVING clauses:
    - Will discuss later on.

Subqueries that return a constant:
- Also referred to as subqueries that return a scalar.
- Most easy case to understand:
  ```sql
  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!
- Next, we focus on subqueries that return relations.

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>
</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>

Conditions involving relations:
- Test that a relation satisfies some condition:
  - WHERE EXISTS (SELECT ...) - TRUE if subquery result is not empty.
  - WHERE UNIQUE (SELECT ...) - TRUE if subquery result has no duplicates.
- Find names of sailors who've reserved boat #103:
  ```sql
  SELECT S.sname
  FROM Sailors S
  WHERE EXISTS (SELECT *
                 FROM Reserves R
                 WHERE R.bid=103 AND S.sid=R.sid)
  ```
- Subquery is CORRELATED with parent query.
Conditions involving relations and tuples

- Typically have some sort of set operations semantics
  - WHERE field \textit{IN} (SELECT …)
  - WHERE field \textit{op} \textit{ANY} (SELECT …)
  - WHERE field \textit{op} \textit{ALL} (SELECT …)

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

Find names of sailors who've reserved boat #103

```
SELECT S.sname
FROM Sailors S
WHERE S.rating > \textit{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

```
SELECT S.sname
FROM Sailors S
WHERE S.age \textit{\geq} \textit{ALL} (SELECT S1.age
  FROM Sailors S1)
```

Find names of sailors with maximum age

Subqueries in the FROM clause

```
SELECT SQ.sname, SQ.bname
FROM ( SELECT S.sname, B.name AS bname
  FROM Sailors S, Boats B, Reserves R
  WHERE S.sid=R.sid AND
  B.bid=R.bid ) SQ
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:

```
SELECT S.sid
FROM Sailors S, Boats B, Reserves R
WHERE S.sid=R.sid AND R.bid=B.bid AND B.color='red'
  AND S.sid \textit{IN} (SELECT R2.sid
  FROM Sailors S2, Boats B2, Reserves R2
  WHERE S2.sid=R2.sid AND R2.bid=B2.bid
  AND B2.color='green')
```

Similarly, \textit{EXCEPT} queries re-written using \textit{NOT IN}.

Nested Queries - Review

- Nested queries returning a constant
  - Typically constant is compared with other value in the \textit{WHERE} clause
  - \textit{WHERE} field = (SELECT bid FROM …) …

- Nested queries returning a relation
  - in \textit{WHERE} clause
  - \textit{WHERE EXISTS} (SELECT bid FROM …) …
  - \textit{WHERE field \textit{IN} (SELECT bid FROM …)} …
  - \textit{WHERE field \textit{op} \textit{ANY} (SELECT bid FROM …)} …
  - in \textit{FROM} clause followed by range variable
  - \textit{FROM Sailors, (SELECT bid FROM Boats) Bids …}
Find sids of sailors who’ve reserved only red boats
SELECT R.sid FROM Reserves R, Boats B
WHERE B.bid = R.bid AND B.color = 'red' AND R.sid NOT IN
(SELECT R1.sid FROM Reserves R1, Boats B1
WHERE B1.color <> 'red' AND B1.bid = R1.bid)

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)

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')

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)