SQL Division
**Division**

- Used to answer queries such as:
  
  *Find sailors who have reserved all boats.*

- Let $A$ have 2 fields, $x$ and $y$; $B$ have only field $y$:

  $A/B = \left\{ \langle x \rangle | \exists \langle x, y \rangle \in A \ \forall \langle y \rangle \in B \right\}$

  - $A/B$ contains all $x$ tuples (sailors) such that for every $y$ tuple (boat) in $B$, there is an $xy$ tuple in $A$.
  - Or, if the set of $y$ values (boats) associated with an $x$ value (sailor) in $A$ contains all $y$ values in $B$, the $x$ value is in $A/B$.

- In general, $x$ and $y$ can be any sets of fields (not singletons).
### Examples of Division A/B

<table>
<thead>
<tr>
<th>sid</th>
<th>bid</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>b1</td>
</tr>
<tr>
<td>s1</td>
<td>b2</td>
</tr>
<tr>
<td>s1</td>
<td>b3</td>
</tr>
<tr>
<td>s1</td>
<td>b4</td>
</tr>
<tr>
<td>s2</td>
<td>b1</td>
</tr>
<tr>
<td>s2</td>
<td>b2</td>
</tr>
<tr>
<td>s3</td>
<td>b2</td>
</tr>
<tr>
<td>s4</td>
<td>b2</td>
</tr>
<tr>
<td>s4</td>
<td>b4</td>
</tr>
</tbody>
</table>

#### A/B1

<table>
<thead>
<tr>
<th>sid</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
</tr>
<tr>
<td>s2</td>
</tr>
<tr>
<td>s3</td>
</tr>
<tr>
<td>s4</td>
</tr>
</tbody>
</table>

#### A/B2

<table>
<thead>
<tr>
<th>sid</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
</tr>
<tr>
<td>s4</td>
</tr>
</tbody>
</table>

#### A/B3

<table>
<thead>
<tr>
<th>sid</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
</tr>
</tbody>
</table>
Query 1

“Find the names of sailors who’ve reserved all boats”

\[ \rho (\text{Tempsids}, (\pi_{\text{sid},\text{bid}} \text{Reserves}) \setminus (\pi_{\text{bid}} \text{Boats})) \]

\[ \pi_{\text{sname}} (\text{Tempsids} \bowtie \text{Sailors}) \]
Query 2

“Find sailors who’ve reserved all red boats”

\[ \rho(Temp,(\pi_{sid,bid} Reserves)/(\pi_{bid} (\sigma_{color='red'} Boats)))) \]

\[ \pi_{sname}(Temp \bowtie Sailors) \]
Expressing A/B Using Basic Operators

- For A/B, compute all x values that are not **disqualified** by some y value in B
  - x value is **disqualified** if by attaching y value from B, we obtain an xy tuple that is not in A

**Disqualified x values:**

\[ \pi_x((\pi_x(A) \times B) - A) \]

**A/B:**

\[ \pi_x(A) - \text{all disqualified tuples} \]

\[ \pi_x(A) - \pi_x((\pi_x(A) \times B) - A) \]
Division in SQL

- Not supported as primitive operator

- Need to use nested queries to express division
  - One of the most subtle queries in SQL
  - Need to pay close attention to writing SQL division queries!

- There are two ways of writing division queries
  - Using the set **EXCEPT** operator (2-level nesting)
  - Without the **EXCEPT** operator (3-level nesting)
“Find sailors who’ve reserved all boats.”

With \textbf{EXCEPT}:

\texttt{SELECT S.sname}
\texttt{FROM Sailors S}
\texttt{WHERE NOT EXISTS}
\texttt{(}
\texttt{  (SELECT B.bid FROM Boats B)}
\texttt{  EXCEPT}
\texttt{  (SELECT R.bid FROM Reserves R}
\texttt{    WHERE R.sid=S.sid)}
\texttt{  )}
\texttt{)}


Division: Solution 2

“Find sailors who’ve reserved all boats.”

Without **EXCEPT:**

```
SELECT S.sname
FROM Sailors S  /* Sailors S such that ... */
WHERE NOT EXISTS (SELECT B.bid
FROM Boats B
WHERE NOT EXISTS (SELECT *
FROM Reserves R
WHERE R.bid=B.bid
AND R.sid=S.sid))
```
“Find sailors who’ve reserved all red boats.”

With **EXCEPT**:  
SELECT  S.sname  
FROM  Sailors S  
WHERE  NOT EXISTS  
   (  
      (SELECT  B.bid FROM  Boats B  
         WHERE  B.color = ‘red’)  
      EXCEPT  
      (SELECT  R.bid FROM  Reserves R  
         WHERE  R.sid=S.sid)  
   )
“Find sailors who’ve reserved all red boats.”

Without `EXCEPT`:

```sql
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS (SELECT B.bid
    FROM Boats B
    WHERE B.color='red' AND
    NOT EXISTS (SELECT *
        FROM Reserves R
        WHERE R.bid=B.bid
        AND R.sid=S.sid))
```
Another Example

Find names of actors who star in ALL movies produced by Universal in year 1990.

```
SELECT A.name FROM Actors A
WHERE NOT EXISTS(  
    SELECT M.movie_id FROM Movies M  
    WHERE M.year=1990 AND M.studio='Universal'  
    EXCEPT  
    SELECT S.movie_id FROM Stars_In S  
    WHERE S.actor_id=A.actor_id  
)
```