SQL Notes

• Went through major standardizations which led to its wide acceptance
  – SQL-86 (SQL)
    • Queries, basic definitions & manipulation
  – SQL-89
    • Referential integrity
  – SQL-92
    • Revised & Expanded
  – SQL-99
    • Archive rules & triggers, some recursive operations (?), object-oriented features

REVIEW: Examples (8): A new kind of query

EMP(EMPNO, NAME, DNO, JOB, MGR, SAL, COMMISSION)
DEPT(DNO, DNAME, LOC)

Q11. Find the numbers of those departments that have employees who can do some job that is done by an employee in department D3.

Q12. Find the numbers of those departments that have employees who can do all the jobs that are done by an employee in department D3.

REVIEW: Examples (8): continued (exists, ∃)

Q11. Find the numbers of those departments that have employees who can do some job that is done by an employee in department D3. Answer: D1 and D2

EMP(EMPNO, ..., DNO, JOB, ...)  
100 D3 electrician 
200 D3 plumber 
EMP(EMPNO, ..., DNO, JOB, ...)  
500 D1 plumber 
900 D2 electrician 
πDNO(πJOB(σDNO=D3(EMP)) * EMP)

What about department D3?
REVIEW: Examples (8): continued (for all, ∀)

Q12. Find the numbers of those departments that have employees who can do all the jobs that are done by an employee in department D3. Answer: D1, but not D2

<table>
<thead>
<tr>
<th>EMP</th>
<th>EMPNO</th>
<th>DNO</th>
<th>JOB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
<td>D3</td>
<td>electrician</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>D3</td>
<td>plumber</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>D3</td>
<td>electrician</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>D1</td>
<td>electrician</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>D1</td>
<td>plumber</td>
</tr>
<tr>
<td></td>
<td>600</td>
<td>D1</td>
<td>carpenter</td>
</tr>
<tr>
<td></td>
<td>700</td>
<td>D2</td>
<td>electrician</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>D2</td>
<td>carpenter</td>
</tr>
<tr>
<td></td>
<td>900</td>
<td>D2</td>
<td>electrician</td>
</tr>
</tbody>
</table>

SQL: Examples (8): continued (exists, ∃)

Q11. Find the numbers of those departments that have employees who can do some job that is done by an employee in department D3. Answer: D1 and D2

```
select DNO from DEPT where exists 
  ( select * from EMP ED3 where ED3.DNO = 'D3' 
  and exists 
    ( select * from EMP EY where EY.JOB = ED3.JOB 
    and EY.DNO = DEPT.DNO) 
  and DNO <> 'D3' )
```

The order of the two "selects" does not matter.
Still gets D3.

Eliminate department D3.
**The trick from logic.**

For all $X$ there exists a $Y$ such that $P(X,Y)$ is true. is equivalent to

There does not exist an $X$ for which there does not exist a $Y$ such that $P(X,Y)$ is true.

$$\forall X \exists Y \text{ such that } P(X,Y)$$

is equivalent to

$$\neg \exists X \neg \exists Y \text{ such that } P(X,Y)$$

---

**Q12.** Find the numbers of those departments that have employees who can do all the jobs that are done by an employee in department D3.

**Answer:** D1, but not D2

```sql
SELECT DNO FROM DEPT WHERE ALL ( SELECT * FROM EMP ED3 WHERE ED3.DNO = 'D3' AND EXISTS ( SELECT * FROM EMP EY WHERE EY.JOB = ED3.JOB AND EY.DNO = DEPT.DNO ))
```

However no for all exists in SQL. Use two not exists.

```sql
SELECT DNO FROM DEPT WHERE NOT EXISTS ( SELECT * FROM EMP ED3 WHERE ED3.DNO = 'D3' AND NOT EXISTS ( SELECT * FROM EMP EY WHERE EY.JOB = ED3.JOB AND EY.DNO = DEPT.DNO ))
```
Q12: Find the numbers of those departments that have employees who can do all the jobs that are done by an employee in department D3. Answer: D1, but not D2

```
select DNO
from DEPT
where not exists
  ( select *
    from EMP ED3
    where ED3.DNO = 'D3'
    and not exists
      ( select *
        from EMP EY
        where EY.JOB = ED3.JOB
        and EY.DNO = DEPT.DNO )
  )
and DNO <> 'D3'
```

**SQL: Examples (8): continued (for all, ∀)**

**SQL: for all (∀) using count Function**

**First Attempt:**

```
select DNO
from DEPT D
where ( select count(distinct JOB)
    from EMP ED3
    where ED3.DNO = 'D3'
  ) = ( select count(distinct JOB)
     from EMP EY
     where EY.DNO = D.DNO
  )
```

DJOBS: Jobs done by employees in department D.

D3JOBS: Jobs done by employees in department D3.

**Second Attempt:**

```
select DNO
from DEPT D
where ( select count(distinct JOB)
    from EMP ED3
    where ED3.DNO = 'D3'
  ) = ( select count(distinct EY.JOB)
    from EMP EY, EMP ED3
    where EY.DNO = D.DNO
    and EY.JOB = ED3.JOB
    and ED3.DNO = 'D3'
  )
```

DJOBS: Jobs done by employees in department D that are also done by employees in department D3.

D3JOBS: Jobs done by employees in department D3.
Why does this approach work?

The where clause is applied to a specific department D.

```
select DNO
from DEPT D
where ( select count(distinct JOB)
    from EMP ED3
    where ED3.DNO = 'D3')
= ( select count(distinct EY.JOB)
    from EMP EY, EMP ED3
    where EY.DNO = D.DNO
    and EY.JOB = ED3.JOB
    and ED3.DNO = 'D3')
```

D3JOBS: Jobs done by employees in department D3.

DJOBS: Jobs done by employees in department D that are also done by employees in department D3.

Works if DJOBS ⊆ D3JOBS.

• Why is this a general rule?
• Why is the rule satisfied in this case?