**Database Design, CSCI 340, Spring 2016**
**More Practice with Queries, Feb. 5**

The SPJ database, for Suppliers-Parts-Projects,was created with the statements below (for MySQL add semicolons). A sample extension should be included with this lab.

```
CREATE DATABASE
       SuppliersPartsProjects

/******* S (Supplier) Table */
CREATE TABLE S
(sNo CHAR(4) PRIMARY KEY,
sName CHAR(30),
status INT,
city CHAR(20))


/******* P (Part) Table */
CREATE TABLE P
(pNo CHAR(3) PRIMARY KEY,
pName CHAR(30),
pColor CHAR(20),
weight INT,
city CHAR(20))
```

```
/******* J (Project) Table */
CREATE TABLE J
(jNo CHAR(4) PRIMARY KEY,
jName CHAR(30),
city CHAR(20))

/******* SPJ (Supplier-Part_Project)
Table */
CREATE TABLE SPJ
(sNo CHAR(4) NOT NULL REFERENCES
       S(sNo),
pNo CHAR(3) NOT NULL REFERENCES
       P(pNo),
jNo CHAR(4) NOT NULL REFERENCES
       J(jNo),
qty INT,
PRIMARY KEY (sNo, pNo, JNo))
```

Thanks to C.J. Date, An Introduction to Database Systems, for the database and many of the queries.

1. Looking at the sample extension, tell the number of unique projects which are supplied by supplier S005.

   4

2. Write a query which gives the number of unique projects which are supplied by supplier S005.

```
SELECT COUNT(DISTINCT jNo)
FROM SPJ
WHERE sNo ='S005';
```

3. Get all unique pairs of city names, such that a supplier in the first city supplies a project in the second city.

```
SELECT DISTINCT S.city, J.city
FROM SPJ JOIN S ON SPJ.sNo=S.sNo
        JOIN J ON SPJ.jNo=J.jNo;
```

4. Get the part numbers for parts supplied by a supplier in London. Don't duplicate part numbers. (The result is P01 & P06.)

```
SELECT DISTINCT pNo
FROM SPJ
WHERE sNo IN
        (SELECT sNo
        FROM S
        WHERE city='London');
```

5. For each supplier which supplies parts to a project, tell the total number of parts supplied to some project. (Supplier S001 supplies 900 parts to some project. Supplier S002 supplies 3100 parts to some project, …)

```
SELECT sNo, SUM(qty)
FROM SPJ
GROUP BY sNo;
```

6. For each supplier, tell the total number of parts supplied to some project. This query is very similar to the above except supplier S006, which doesn't supply any parts to any project, must be listed.

```
SELECT S.sNo, ISNULL(SUM(qty),0)
FROM S LEFT OUTER JOIN SPJ
        ON S.sNo=SPJ.sNo
GROUP BY S.sNo;
```

7. Get all supplier name / part name / project name triples such that the indicated supplier, part, and project are all located in the same city. List the name of the city along with the names. Note that the supplier does not need to supply a part for a project. (The result contains 22 records, 4 is Paris and the rest in London.)

```
SELECT sName, pName, jName, S.city
FROM S, P, J
WHERE S.city = P.city AND
      P.city = J.city;
```

8. Repeat the above, only this time the supplier must supply a part for the project (and they the supplier, part and project must all be collocated). (Problematic with MySQL because it asks about intermediate results. In this case just write the query.)

```
SELECT sName, pName, jName, S.city
FROM S JOIN SPJ ON S.sNo=SPJ.sNo
        JOIN P ON P.pNo=SPJ.pNo
                JOIN J ON J.jNo=SPJ.jNo
WHERE S.city = P.city AND
        P.city = J.city;
```

9. List the names of the cites where either a supplier, a part manufacturer, or a project is located. List the names alphabetically and don't duplicate names.

```
(SELECT DISTINCT city
FROM S)
UNION
(SELECT DISTINCT city
FROM P)
UNION
(SELECT DISTINCT city
FROM J)
ORDER BY city;
```

10. Get project numbers for projects not supplied with any red part or by any London supplier. (Note that MySQL does not support the MINUS operator but does allow NOT IN.)

```
SELECT jNo
FROM J
WHERE jNo NOT IN
                        (SELECT jNo
                         FROM SPJ JOIN P ON SPJ.pNo=P.pNo
                         WHERE pColor='Red')
AND jNo NOT IN
                        (SELECT jNo
                         FROM SPJ JOIN S ON SPJ.sNo=S.sNo
                         WHERE city='London') ;
```