Greg Matthews
CS 500
February 11th, 2018

HW 3: SQL

Part 1 (60 points):

(a). For each job, list all candidates who have at least two of the necessary skills for that job. Result should have the schema (job, grade, candidate, cnt), where cnt is the number of skills a candidate has for a job.

```
select job, grade, candidate, count(*) as cnt
from Candidates_Skills C_S, Jobs_Skills J_S
where J_S.skill = C_S.skill
group by job, grade, candidate
having count(*) > 1;
```

(b). For each position, list all candidates whose salary requirements meet the salary budget. List candidates in order of required salary, from lower to higher. Result should have the schema(job, grade, company, candidate, candidate_salary).

```
select job, grade, company, candidate,
P.salary as salary, C.salary as candidate_salary
from Candidates C, Positions P
where C.salary < P.salary
order by C.salary;
```

(c). Are there job skills with no qualified candidates in our database? A candidate who has the skill but not the required number of years of experience is not qualified for the skill. The query should list all such skills, with the schema (skill, years).

```
 select distinct skill, years
from Jobs_Skills J_S
EXCEPT
select distinct J_S.skill, J_S.years
from Candidates_Skills C_S, Jobs_Skills J_S
where C_S.years >= J_S.years
and C_S.skill = J_S.skill;
```

(d). For each skill (e.g., Java or DB), compute the maximum and the average number of years of experience among the candidates who have that skill, and the total number of candidates who have that skill in our database. Round the average number of years to 1 digit after the decimal point. Result should have the schema (skill, max_years, avg_years, cnt).

```
select skill, MAX(years) as max_years,
round(AVG(years),1) as avg_years, count(*) as cnt
from Candidates_Skills C_S
group by skill;
```

(e). List pairs of candidates such that candidate 1 has more years of combined Java and DB experience but a lower salary requirement than candidate 2. The result should have the schema (candidate1, candidate2).

```
 select C_1.candidate as candidate1, C_2.candidate as candidate2
from Candidates as C_1, Candidates as C_2,
Candidates_Skills as C_11, Candidates_Skills as C_12,
Candidates_Skills as C_21, Candidates_Skills as C_22
where C_1.candidate = C_11.candidate
and C_1.candidate = C_12.candidate
and C_2.candidate = C_21.candidate
and C_2.candidate = C_22.candidate
and C_11.skill = 'Java' and C_12.skill = 'DB'
and C_21.skill = 'Java' and C_22.skill = 'DB'
and C_11.year + C_12.year > C_21.year + C_22.year
and C_1.salary < C_2.salary;
```

(f). List pairs of candidates who both have at least 3 years of DB experience. Return each pair of candidates exactly once, i.e., do not return both ('Ann', 'Cathy') and ('Cathy', 'Ann'). Do not return a pair that contains the same candidate twice, e.g., do not return ('Ann', 'Ann'). Result should have the schema (candidate1, candidate2).

```
select C_1.candidate as candidate1, C_2.candidate as candidate2
from Candidates_Skills as C_1, Candidates_Skills as C_2
where C_1.candidate > C_2.candidate
and C_1.skill = 'DB' and C_2.skill = 'DB'
and C_1.years >= 3 and C_2.years >= 3;
```

Part 2 (60 points):

(a). For each country, compute the number of years in which one of its tennis players was ranked first. Result should have the schema (country, num_years).

```
select C.name as country, count(Y_R_F.year) as num_years
from Tennis_Players T_P, Years_Ranked_First Y_R_F, Countries C
where T_P.name = Y_R_F.name
and T_P.country = C.name
group by C.name;
```

(b). List pairs of tennis players (player1, player2) in which player1 both has a lower (better) ATP rank than player 2 and comes from a less populous country.

```
select TP1.name as player1, TP2.name as player2
from Tennis_Players TP1, Tennis_Players TP2, Countries C1,Countries C2
where TP1.country = C1.name
and TP2.country = C2.name
and TP1.ATP_rank < TP2.ATP_rank
and C1.population > C2.population;
```

(c). List pairs of players from the same country. List each pair exactly once. That is, you should list either (Djokovic, Raonic, Serbia) or (Raonic, Djokovic, Serbia), but not both. Result should have the schema (player1, player2, country).

```
select TP1.name as player1, TP2.name as player2, C.name as country
from Tennis_Players TP1, Tennis_Players TP2, Countries C
where TP1.name > TP2.name
and TP1.country = C.name
and TP2.country = C.name;
```

(d). For countries with at least 2 tennis players, list country name, GDP and average age of its tennis players. Result should have the schema (country, GDP, avg_age).

```
select C.name as country, C.gdp, AVG(TP.age) as avg_age
from Countries C, Tennis_Players TP
where TP.country = C.name
group by C.name, gdp
having count(*) > 1;
```

(e). List country name, GDP and population of each country. For countries that have tennis players in our database, also list the minimum age of its tennis players. Result should have the schema (country, GDP, population, min_age).

```
select C.name as country, C.gdp, population, MIN(TP.age) as min_age
from Countries C, Tennis_Players TP
where TP.country = C.name
group by C.name, C.gdp;
```

(f). List names of countries who had a top-ranked tennis player both in 2010 or earlier (i.e., between 2004 and 2010, inclusive) and after 2010 (i.e., between 2011 and 2015, inclusive).

```
select C.name
from Countries C, Years_Ranked_First YRF, Tennis_Players TP
where C.name = TP.country
and TP.name = YRF.name
and YRF.year <= 2010 and YRF.year >= 2004
INTERSECT
select C.name
from Countries C, Years_Ranked_First YRF, Tennis_Players TP
where C.name = TP.country
and TP.name = YRF.name
and YRF.year >= 2010 and YRF.year <=2015;
```