HW3

Thursday, March 3, 2022 3:58 PM

Information and Database Management Systems I

(CIS 4301)

Spring 2022

Instructor: Dr. Markus Schneider

TA: Kyuseo Park

Homework 3

Name:	Charles Richardson
UFID:	73112398
Email Address:	crichardson5@ufl.edu

Pledge (Must be signed[1] according to UF Honor Code)

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

Student signature

[1]Each student is obliged to print out this page, fill in the requested information in a handwritten and readable manner, make the *handwritten* signature, scan this page into PDF, and put this page as the first page of the PDF submission.

QUESTION 1

a) Part A

1. According to the internet, the statement **INITIALLY DEFERRED DEFERRABLE** means that constraints do not apply until the table is committed.
2. This command is used in the assignment because of the magnitude of the dataset. With such a large dataset, all of the constraints should not be checked until the tables have been crested (modified). This results in shorter wait times for database creation and modification.

b) Part B

1.

```
SELECT country, province, COUNT(island) AS islandCount
FROM locatedOn
GROUP BY country, province
ORDER BY islandCount DESC
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 253 in 0.092 seconds

	COUNTRY	PROVINCE	ISLANDCOUNT
1	GB	South East	15
2	RC	Taiwan	14
3	GB	North West	14
4	RP	Metro Manila	13
5	GB	East of England	10
6	GB	Yorkshire and the Humber	9
7	GB	South West	8
8	GB	West Midlands	8
9	RI	Jawa Barat	8
10	RH	Haiti	7
11	RM	Madagascar	7
12	GB	Scotland	6
13	NZ	New Zealand	5
14	J	Kanagawa	5
15	J	Chiba	5
16	GB	East Midlands	5
17	E	Canarias	4
18	IRL	Ireland	4
19	RI	Jawa Timur	4
20	J	Osaka	4
21	J	Hyogo	4
22	RI	Banten	4
23	GB	North East	4
24	IS	Iceland	4
25	I	Sicilia	4

26	RI	Jawa Tengah	4
27	DK	Hovedstaden	4
28	PR	Puerto Rico	4
29	RI	Sumatera Utara	3
30	BR	São Paulo	3
31	MAL	Sabah	3
32	USA	New York	3
33	CY	Cyprus	3
34	GB	Wales	3
35	J	Hokkaido	3
36	J	Tokyo	3
37	CN	Hainan	2
38	USA	Hawaii	2

2.

```
SELECT name,
       ROUND(population / area, 4) AS "Population Density",
       ROUND(population / (SELECT SUM(population) FROM Country), 10) * 100 AS "Percentage"
FROM Country
ORDER BY "Population Density" desc, "Percentage" desc
FETCH FIRST 10 ROWS ONLY
```

Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.025 seconds

	NAME	Population Density	Percentage
1	Macao	34531.4375	0.00779449
2	Monaco	19392.1053	0.00051979
3	Singapore	8025.1344	0.07162004
4	Melilla	6539.6667	0.00110711
5	Hong Kong	6475.8022	0.09976295
6	Gaza Strip	5203.537	0.02679443
7	Gibraltar	5011.8462	0.00045958
8	Ceuta	4576.4444	0.00116213
9	Bahrain	1991.2839	0.01741718
10	Holy See	1913.6364	0.00001188

3.

```
SELECT name
FROM located,
     country
WHERE   located.country = country.code and
        lake = (SELECT name
                FROM lake
                WHERE area = (SELECT MAX(area) FROM lake))
GROUP BY name
```

Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.038 seconds

	NAME
1	Kazakhstan

2	Iran
3	Russia
4	Azerbaijan

4.

```
SELECT name, length
FROM River, RiverThrough
WHERE River.name = RiverThrough.river and
      sea = 'Atlantic Ocean'
ORDER BY length desc
FETCH FIRST 2 ROWS ONLY
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.022 sec

	NAME	LENGTH
1	Zaire	4374
2	Niger	4184

5.

```
SELECT name, area
FROM island, geo_island, encompasses, locatedOn
WHERE continent = 'Africa' AND
      geo_island.country = encompasses.country AND
      locatedOn.province = geo_island.province AND
      locatedOn.island = island.name AND
      area > 1000
GROUP BY name, area
ORDER BY area DESC
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.066 seconds

	NAME	AREA
1	Madagaskar	587041
2	Reunion	2510
3	Bioko	2017

4	Mauritius	1860
5	Sansibar	1658
6	Grand Comoro	1148

6.

```
SELECT c1, n1, dependent as c2, COUNT(dependent) as n2, n1 - count(dependent) as "difference
FROM politics, (SELECT wasdependent AS c1, COUNT(wasdependent) as n1
                FROM politics
                WHERE wasdependent IS NOT NULL
                GROUP BY wasdependent
                ORDER BY COUNT(wasdependent) DESC
                FETCH FIRST ROW ONLY)
WHERE dependent IS NOT NULL
GROUP BY c1, n1, dependent
ORDER BY COUNT(wasdependent) DESC, COUNT(dependent) DESC
FETCH FIRST ROW ONLY
```

Script Output × Query Result ×

SQL | All Rows Fetched: 1 in 0.028 seconds

	C1	N1	C2	N2	difference
1	GB	55	GB	13	42

7.

```
SELECT name
FROM Economy, Country
WHERE agriculture > 50 and
      country = code
```

Script Output × Query Result ×

SQL | All Rows Fetched:

	NAME
1	Comoros
2	Falkland Islands
3	Guinea-Bissau
4	Liberia
5	Central African Republic
6	Somalia

Worksheet | Query Builder

8.

```
SELECT name, continent
FROM City,
        (SELECT continent, max(latitude) as lat
        FROM City, encompasses
        WHERE City.country = encompasses.country and
            continent != 'Asia'
        GROUP BY continent)
WHERE City.latitude = lat
ORDER BY latitude DESC
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.022 seconds

	NAME	CONTINENT
1	Longyearbyen	Europe
2	Nuuk	America
3	Annaba	Africa
4	Antalya	Africa
5	Saipan	Australia/Oceania

9.

```
SELECT Country.name, GDP
FROM Religion, isMember, Economy, Country
WHERE Religion.name = 'Muslim' and
      organization = 'NATO' and
      isMember.country = Religion.country and
      Economy.country = Religion.country and
      Country.code = Religion.country and
      percentage > 5
```

Script Output | Query Result

SQL | All Rows Fetched: 8 in 0.25 seconds

	NAME	GDP
1	Albania	12800
2	Belgium	507400
3	Bulgaria	53700
4	France	2739000

5	Germany	3593000
6	Montenegro	4518
7	Netherlands	722300
8	Turkey	821800

10.

```
SELECT r as religion, ROUND(SUM(total),0) as believers
FROM (SELECT religion.name AS r, percentage * population as total
      FROM religion, country
      WHERE religion.country = country.code)
GROUP BY r
ORDER BY believers DESC
FETCH FIRST 5 ROWS ONLY
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.02 seconds

	RELIGION	BELIEVERS
1	Muslim	168958599331
2	Hindu	102677473828
3	Roman Catholic	99370849706
4	Protestant	40700314958
5	Buddhist	30760171782

11.

```
SELECT province
FROM  Economy, geo_island
WHERE GDP > 1000000 and
      economy.country = geo_island.country
GROUP BY province HAVING count(island) > 2
```

Script Output | Query Result

SQL | All Rows Fetched: 13 in 0.026 sec

	PROVINCE
1	Scotland
2	California
3	Ontario

4	Canarias
5	Illes Balears
6	New York
7	Schleswig-Holstein
8	Hawaii
9	Calabria
10	Nunavut
11	Sicilia
12	Sakhalin
13	Niedersachsen

12.

```
SELECT name
FROM encompasses, country
WHERE percentage < 100 and
      country = code
GROUP BY name
```

Script Output | Query Result

SQL | All Rows Fetched

	NAME
1	Egypt
2	Indonesia
3	Kazakhstan
4	Russia
5	Turkey

```
SELECT continent, max(elevation)
FROM geo_mountain, mountain, encompasses
WHERE mountain.name = geo_mountain.mountain and
      encompasses.country = geo_mountain.country
GROUP BY continent
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.044 seconds

13.

	CONTINENT	MAX(ELEVATION)
1	Asia	8848
2	Europe	7010
3	Africa	5895
4	America	6962
5	Australia/Oceania	4884

14.

```
SELECT name, d, e
FROM Country, (SELECT geo_mountain.country AS c, max(sea.depth) AS d, max(mountain.elevation) AS e
               FROM geo_mountain, geo_sea, Mountain, Sea
               WHERE geo_mountain.country = geo_sea.country and
                geo_mountain.province = geo_sea.province and
                geo_mountain.mountain = Mountain.name and
                geo_sea.sea = Sea.name
               GROUP BY geo_mountain.country HAVING max(mountain.elevation) > max(sea.depth))
WHERE Country.code = c
```

Script Output × Query Result ×

SQL | All Rows Fetched: 6 in 0.036 seconds

	NAME	D	E
1	Bulgaria	2211	2925
2	Finland	459	1365
3	Georgia	2211	5200
4	Iran	3350	3941
5	Sweden	459	2099
6	Turkey	2211	3937

15.

```
SELECT SUM(Length) as "Shared Border Length"
FROM Borders, Country
WHERE Country.code = Borders.country1 and
country.name = 'China'
```

Query Result ×

SQL | All Rows Fetched: 1 in 0.021 secc

	Shared Border Length
1	21597.34

```
SELECT organization -- All organizations
FROM isMember, encompasses
MINUS
SELECT organization --Org with non-Asian members
FROM isMember, encompasses
WHERE encompasses.continent <> 'Asia' and
```

16.

```
        encompasses.country = isMember.country
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 3 in 0.641 seconds

	ORGANIZATION
1	BIMSTEC
2	GCC
3	SACEP

17.

```
SELECT SUM(A) as top10,
       SUM(B) as rest_world,
       (SUM(A) - SUM(B)) as difference
FROM (SELECT AREA as A
       FROM country
       ORDER BY area DESC
       FETCH FIRST 10 ROWS ONLY),
(SELECT AREA as B
       FROM country
       ORDER BY area DESC
       OFFSET 10 ROWS
       FETCH NEXT 100 PERCENT ROWS ONLY)
```

Script Output | Query Result | Query Result

SQL | All Rows Fetched: 1 in 0.022 seco

	TOP10	REST_WORLD	DIFFERENCE
1	17170550046	621860736.4	16548689309.6

18.

```
SELECT country.name
FROM Country, City
WHERE Country.capital = City.name and
      City.latitude > 0 and
      City.population < 1000
```

Script Output | Query Result | Query R

SQL | All Rows Fetched: 4 in 0.02 s

	NAME
1	Monaco
2	Holy See
3	Montserrat
4	Palau

19.

```
SELECT name
FROM lake, geo_lake
WHERE elevation > (SELECT AVG(elevation)
                   FROM lake) and
      lake.name = geo_lake.lake and
      geo_lake.country = 'USA'
GROUP BY name
```

Script Output × | Query Result × | Query R

SQL | All Rows Fetched: 6 in 0.043

	NAME
1	Great Salt Lake
2	Mazama Crater Lake
3	Lake Tahoe
4	Pyramid Lake
5	Lake Powell
6	Mono Lake

20.

```
SELECT River, Country, Count(*) as "CROSSED PROVS"
FROM Geo_River
GROUP BY River, Country HAVING COUNT(*) > 11
```

Script Output × | Query Result × | Query Result 1 ×

SQL | All Rows Fetched: 2 in 0.019 seconds

	RIVER	COUNTRY	CROSSED PROVS
1	Donau	RO	12
2	Volga	R	13

21.

```
SELECT country.province, mountain, ROUND(population/area, 0) as density
FROM Mountain, geo_mountain, country
WHERE type = 'volcano' and
      mountain.name = geo_mountain.mountain and
      geo_mountain.country = country.code and
      geo_mountain.province = country.province
ORDER BY density DESC
FETCH FIRST ROW ONLY
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 1 in 0.04 seconds

	PROVINCE	MOUNTAIN	DENSITY
1	South Korea	Halla-San	521

```
SELECT country, ROUND(country.population * POWER(1 + (population_growth / 100),5), 0) as "Population after 5 years",
    RANK() OVER (ORDER BY country.population * POWER(1 + (population_growth / 100),5) DESC) as Rank
FROM population, Country
WHERE population.country = country.code and
      country.population IS NOT NULL and
      population_growth IS NOT NULL
FETCH FIRST 5 ROWS ONLY
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 5 in 0.024 seconds

	COUNTRY	Population after 5 years	RANK
1	CN	1390920437	1
2	IND	1288449172	2
3	USA	331323564	3
4	RI	264330084	4
5	PK	223724547	5

22.

```
SELECT country, ROUND(country.population * POWER(1 + (population_growth / 100),5), 0) as "Population after 5 years",
    RANK() OVER (ORDER BY country.population * POWER(1 + (population_growth / 100),5) DESC) as Rank
FROM population, Country
WHERE population.country = country.code and
      country.population IS NOT NULL and
      population_growth IS NOT NULL
FETCH FIRST 5 ROWS ONLY
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 5 in 0.024 seconds

	COUNTRY	Population after 5 years	RANK
1	CN	1390920437	1
2	IND	1288449172	2
3	USA	331323564	3
4	RI	264330084	4
5	PK	223724547	5

23.

```
SELECT country.name
FROM country, (--Countries with more than 3 rivers
    SELECT country as c
    FROM geo_river
```

```
    FROM geo_river
    GROUP BY country HAVING count(*) > 3
    INTERSECT
    --Countries with lakes next to more than 3 provinces
    SELECT country as c
    FROM  geo_lake
    GROUP BY country HAVING COUNT(*) > 3)
WHERE c = Country.code
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 25 in 0.017 seconds

	NAME
1	Argentina
2	Australia
3	Bolivia
4	Brazil
5	Canada
6	China
7	Cote d'Ivoire
8	Ethiopia
9	Finland
10	Germany
11	Ghana
12	Hungary
13	Iran
14	Italy
15	Kazakhstan
16	Russia
17	Sweden
18	Switzerland
19	Tanzania
20	Turkey
21	Uganda
22	Ukraine
23	United States
24	Zaire
25	Zambia

```
SELECT name, length
FROM  river, geo_river, encompasses
WHERE encompasses.continent = 'America' and
encompasses.country = geo_river.country and
geo_river.river = river.name
GROUP BY name, length
```

24.

```
ORDER BY length DESC
FETCH FIRST ROW ONLY
```

Script Output × | Query Result × | Query Result 1

SQL | All Rows Fetched: 1 in 0.025 secon

	NAME	LENGTH
1	Missouri	4130

25.

```
SELECT country.name
FROM country, (SELECT religion.country AS c
               FROM religion
               GROUP BY religion.country HAVING
                COUNT(religion.name) > 4 AND
                MAX(religion.percentage) > 80)
WHERE c = country.code
```

Script Output × | Query Result × | Query Result 1 ×

SQL | All Rows Fetched: 3 in 0.021 seconds

	NAME
1	Indonesia
2	Ukraine
3	Italy

QUESTION 2

A) Draw By Query

1.

Customer	customerID	name	address
	_p	P.	P.

Book	bID	cID	fnumber	customerID	seat	date
				¬_p		

Customer	customerID	name	address

2.

Customer	customerID	name	address
	_p	P.	

Book	bID	cID	fnumber	customerID	seat	date
			_f	_p		

Flight	cID	fnumber	departure	arrival	price:int	seats:int
		_f		_a		

Conditions
_a = "Boston" **or** _a = "Seattle"

3.

Customer	customerID	name	address
	_p	P.	

Book	bID	cID	fnumber	customerID	seat	date
			_f	_p		

Flight	cID	fnumber	departure	arrival	price:int	seats:int
		_f		_a		
		¬_f		_a		

Conditions
_a = "New York"

4.

Customer	customerID	name	address

	_p	P.G._n	

Book	bID	cID	fnumber	customerID	seat	date
		_c		_p		

Company	cID	name	location
	_c	_n1	
		_n2	

Conditions
CNT.UN.ALL._n1 = CNT.UN.ALL._n2

5.

Customer	customerID	name	address
	_c	_n	_a

Book	bID	cID	fnumber	customerID	seat	date
			_f1	_c		_d

Flight	cID	fnumber	departure	arrival	price:int	seats:int
		_f2	_p1	_p2		

CustomerCheck	name	address	fnumber
I.	_n	_a	_f

Conditions
_d = 04/30/22 _p1 = "New York" _p2 = "LA" _f1 = _f2

6.

Company	cID	name	location
	_c1	P._n	

Flight	cID	fnumber	departure	arrival	price:int	seats:int
	_c1					_s
¬	_c2					>_s