LEDE PROGRAM: DATA AND DATABASES DAY 3

Here's my summary of yesterday's PSQL fun in class.

First we checked out the economy table using \d and then SELECT*

```
 \d economy
                          Table "public.economy"
    Column    |         Type          | Collation | Nullable | Default
--------------+-----------------------+-----------+----------+---------
 country      | character varying(4)  |           | not null |
 gdp          | numeric               |           |          |
 agriculture  | numeric               |           |          |
 service      | numeric               |           |          |
 industry     | numeric               |           |          |
 inflation    | numeric               |           |          |
 unemployment | numeric               |           |          |
Indexes:
    "economykey" PRIMARY KEY, btree (country)
Check constraints:
    "economygdp" CHECK (gdp >= 0::numeric)

 SELECT * FROM economy;
 country |   gdp    | agriculture | service | industry | inflation | unemployment
---------+----------+-------------+---------+----------+-----------+--------------
 AL      |    12800 |        19.5 |    68.5 |       12 |       1.7 |         16.9
 GR      |   243300 |         3.5 |    80.5 |       16 |      -0.8 |         27.9
 MK      |    10650 |        10.2 |    62.3 |     27.5 |       2.8 |         28.6
 SRB     |    43680 |         7.9 |    60.3 |     31.8 |       2.2 |         20.1
 MNE     |     4518 |         0.8 |    87.9 |     11.3 |         4 |         19.1
 KOS     |     7150 |        12.9 |    64.5 |     22.6 |       1.8 |         30.9
 AND     |     4800 |          14 |       6 |       79 |       1.1 |            4
 F       |  2739000 |         1.9 |    79.4 |     18.7 |       0.9 |         10.2
 E       |  1356000 |         3.1 |    70.8 |       26 |       1.8 |         26.3
 A       |   417900 |         1.6 |    69.8 |     28.6 |       2.1 |          4.9
 CZ      |   194800 |         2.4 |    60.3 |     37.3 |       1.4 |          7.1
 D       |  3593000 |         0.8 |      69 |     30.1 |       1.6 |          5.3
 H       |   130600 |         3.4 |    68.7 |       28 |       1.9 |         10.5
 I       |  2068000 |           2 |    73.5 |     24.4 |       1.2 |         12.4
 FL      |     5113 |           8 |      55 |       37 |      -0.7 |          2.3
 SK      |    96960 |         3.1 |      67 |     30.8 |       1.7 |         14.4
 SLO     |    46820 |         2.8 |    68.3 |     28.9 |       1.8 |         13.1
 CH      |   646200 |         0.7 |    72.5 |     26.8 |      -0.4 |          3.2
 BY      |    69240 |         9.2 |    44.7 |     46.2 |        19 |            1
 LV      |    30380 |         4.9 |    69.4 |     25.7 |       0.2 |          9.8
 LT      |    46710 |         3.7 |      68 |     28.3 |       1.2 |         12.4
 PL      |   513900 |           4 |    62.7 |     33.3 |         1 |         10.3
```

We answered homework.

```
mondial2=# SELECT country, gdp FROM economy ORDER BY gdp DESC NULLS LAST LIMIT 10;
 country |   gdp
---------+----------
 USA     | 16720000
 CN      |  9330000
 J       |  5007000
 D       |  3593000
 F       |  2739000
 GB      |  2490000
 BR      |  2190000
 R       |  2113000
 I       |  2068000
 CDN     |  1825000
(10 rows)

mondial2=# SELECT country, gdp FROM economy WHERE gdp < 20000;
 country |  gdp
---------+-------
 AL      | 12800
 MK      | 10650
 MNE     |  4518
 KOS     |  7150
 AND     |  4800
 FL      |  5113
 BIH     | 18870
 FARX    |  2320
 MC      |  5748
 GBZ     |  1106
 GBG     |  2742
 IS      | 14590
 RSM     |  1866
 GBJ     |  5100
 M       |  9541
 GBM     |  4076
 MD      |  7932
 TAD     |  8513
 ARM     | 10440
 GE      | 15950
 BHT     |  2133
 BRU     | 16560
 LAO     | 10100

mondial2=# SELECT country, inflation FROM economy ORDER BY inflation DESC NULLS LAST;
 country | inflation
---------+-----------
 SYR     |      59.1
 YV      |      56.2
 IR      |      42.3
 MW      |      26.9
 SUD     |        25
 RA      |      20.8
 BY      |        19
 WEST    |        14
 ER      |        13
 MH      |      12.9
 RG      |      11.9
 YE      |      11.8
 WAL     |      11.1
 BHT     |        11
 GH      |        11
 UZB     |      10.1
 IND     |       9.6
 JA      |       9.4
 BI      |       9.3
```

This wasn't in class, but it might be helpful--here is a very simple JOIN to get country names.

```
SELECT country.name, economy.gdp
FROM economy JOIN country ON economy.country = country.code
ORDER BY economy.gdp DESC NULLS LAST LIMIT 10;
      name      |   gdp
----------------+----------
 United States  | 16720000
 China          |  9330000
 Japan          |  5007000
 Germany        |  3593000
 France         |  2739000
 United Kingdom |  2490000
 Brazil         |  2190000
 Russia         |  2113000
 Italy          |  2068000
 Canada         |  1825000
(10 rows)
```

And here are the columns I am using to join the two tables: country.code and economy.country

```
 SELECT code from country limit 5;
 code
------
 AL
 GR
 MK
 SRB
 MNE
```

```
(5 rows)

 select country from economy limit 5;
 country
---------
 AL
 GR
 MK
 SRB
 MNE
(5 rows)
```

Here we got countries that have the majority of their GDP from agriculture. Here being defined as 50% or more

```
mondial2=# SELECT country, agriculture FROM economy WHERE agriculture > 50;
 country | agriculture
---------+-------------
 FALK    |          95
 RCA     |        56.6
 COM     |          51
 LB      |        76.9
 SP      |        59.3
 GNB     |          58
(6 rows)
```

Here is another definition of majority agriculture. Note the AND in the WHERE test.

```
mondial2=# SELECT country, agriculture FROM economy
mondial2-# WHERE agriculture > service AND agriculture > industry;
 country | agriculture
---------+-------------
 SLB     |          50
 RMM     |        38.5
 ZRE     |        44.3
 RCA     |        56.6
 TCH     |        46.3
 COM     |          51
 LB      |        76.9
 ETH     |          47
 SP      |        59.3
 GNB     |          58
 WAL     |        47.9
(11 rows)
```

In class we noted that the Falkland Islands disappeared, because there was a null value involved. To deal with null values you can use coalesce() to impute values.

```
mondial2=# SELECT country, agriculture FROM economy
mondial2-# WHERE agriculture > coalesce(service, 0) AND agriculture > coalesce(industry,0);
 country | agriculture
---------+-------------
 SLB     |          50
 FALK    |          95
 RMM     |        38.5
 ZRE     |        44.3
 RCA     |        56.6
 TCH     |        46.3
 COM     |          51
 LB      |        76.9
 ETH     |          47
 SP      |        59.3
 GNB     |          58
 WAL     |        47.9
(12 rows)
```

Here's the final answer to part one, with the join on country.code and economy. country

```
SELECT country.name, unemployment FROM economy
mondial2-# JOIN country ON country.code = economy.country ORDER BY economy.unemployment DESC NULLS LAST;
               name                | unemployment
-----------------------------------+--------------
 Zimbabwe                          |           95
 Nauru                             |           90
 Liberia                           |           85
 Burkina Faso                      |           77
 Turkmenistan                      |           60
 Djibouti                          |           59
 Congo                             |           53
 Senegal                           |           48
 Nepal                             |           46
 Bosnia and Herzegovina            |         44.3
 Haiti                             |         40.6
 Swaziland                         |           40
 Kenya                             |           40
 Marshall Islands                  |           36
 Yemen                             |           35
 Afghanistan                       |           35
 Grenada                           |         33.5
 Kosovo                            |         30.9
 Cameroon                          |           30
 Mauritania                        |           30
 Mali                              |           30
 Libya                             |           30
 American Samoa                    |         29.8
 North Macedonia                   |         28.6
 Maldives                          |           28
 Greece                            |         27.9
 Namibia                           |         27.4
 Spain                             |         26.3
 Lesotho                           |           25
 South Africa                      |         24.9
 Nigeria                           |         23.9
 Dominica                          |           23
 Equatorial Guinea                 |         22.3
 Micronesia                        |           22
 Croatia                           |         21.6
 Cape Verde                        |           21
 Gabon                             |           21
```

To demonstrate exactly what is happening in a JOIN I showed the entire 'country' table.

```
mondial2=# SELECT * FROM country;
               name                | code |       capital        |             province              |   area   | po
-----------------------------------+------+----------------------+-----------------------------------+----------+---
 Albania                           | AL   | Tirana               | Albania                           |    28750 |
 Greece                            | GR   | Athina               | Attikis                           |   131940 |   1
 North Macedonia                   | MK   | Skopje               | North Macedonia                   |    25333 |
 Serbia                            | SRB  | Beograd              | Serbia                            |    77474 |
 Montenegro                        | MNE  | Podgorica            | Montenegro                        |    14026 |
 Kosovo                            | KOS  | Prishtine            | Kosovo                            |    10887 |
 Andorra                           | AND  | Andorra la Vella     | Andorra                           |      450 |
 France                            | F    | Paris                | ÃŽle-de-France                     |   547030 |
 Spain                             | E    | Madrid               | Madrid                            |   504750 |   4
 Austria                           | A    | Wien                 | Wien                              |    83850 |
 Czech Republic                    | CZ   | Praha                | Praha                             |    78703 |   1
 Germany                           | D    | Berlin               | Berlin                            |   356910 |   8
 Hungary                           | H    | Budapest             | Budapest                          |    93030 |
 Italy                             | I    | Roma                 | Lazio                             |   301230 |   5
 Liechtenstein                     | FL   | Vaduz                | Liechtenstein                     |      160 |
 Slovakia                          | SK   | Bratislava           | BratislavskÃ½                      |    48845 |
 Slovenia                          | SLO  | Ljubljana            | Slovenia                          |    20256 |
 Switzerland                       | CH   | Bern                 | Bern                              |    41290 |
 Belarus                           | BY   | Minsk                | Minsk City                        |   207600 |
 Latvia                            | LV   | RÄ«ga                 | Latvia                            |    64100 |
 Lithuania                         | LT   | Vilnius              | Lithuania                         |    65200 |
```

Then I joined every column on both tables so you can see what a JOIN is doing. We usually just select the columns we want from each table, but what is happening here is we are essentially building a new combined table based on matching country codes.

```
SELECT * FROM economy
mondial2-# JOIN country ON country.code = economy.country ORDER BY economy.unemployment DESC NULLS LAST;
 country |   gdp    | agriculture | service | industry | inflation | unemployment |               name
---------+----------+-------------+---------+----------+-----------+--------------+-------------------------------
 ZW      |    10480 |        20.1 |    54.5 |     25.4 |       8.5 |           95 | Zimbabwe
 NAU     |      100 |         6.1 |    60.8 |       33 |      -3.6 |           90 | Nauru
 LB      |     1977 |        76.9 |    17.7 |      5.4 |       5.2 |           85 | Liberia
 BF      |    12130 |        33.6 |    42.8 |     23.6 |       2.1 |           77 | Burkina Faso
 TM      |    40560 |         7.2 |    68.4 |     24.4 |         9 |           60 | Turkmenistan
 DJI     |     1459 |           3 |    79.7 |     17.3 |       2.5 |           59 | Djibouti
 RCB     |    14250 |         3.3 |    22.9 |      73.9 |        1.7 |            53 | Congo
 SN      |    15360 |        14.9 |    62.4 |      22.7 |        0.8 |            48 | Senegal
 NEP     |    19340 |        36.8 |    48.7 |      14.5 |        6.7 |            46 | Nepal
 BIH     |    18870 |         8.1 |    65.5 |      26.4 |        0.2 |          44.3 | Bosnia and Herzegovina
 RH      |     8287 |        24.1 |      56 |      19.9 |        6.3 |          40.6 | Haiti
 SD      |     3807 |         7.6 |    44.6 |      47.8 |        6.1 |            40 | Swaziland
 EAK     |    45310 |        29.3 |    53.3 |      17.4 |        5.8 |            40 | Kenya
 MH      |      193 |        14.3 |    71.8 |      13.9 |       12.9 |            36 | Marshall Islands
 YE      |    43890 |         7.7 |    61.4 |      30.9 |       11.8 |            35 | Yemen
 AFG     |    20650 |          20 |    54.4 |      25.6 |        6.8 |            35 | Afghanistan
 WG      |      811 |         5.6 |    78.5 |      15.8 |        2.4 |          33.5 | Grenada
 KOS     |     7150 |        12.9 |    64.5 |      22.6 |        1.8 |          30.9 | Kosovo
```

We jumped into language, and talked about the flaw of showing the top 10 when there are actually 33 countries with 100 of a single language. Here are a few versions of doing the query.

```
                                                              ^
mondial2=# SELECT name, country, percentage FROM language ORDER BY percentage DESC NULLS LAST, country LIMIT 10;
  name   | country | percentage
---------+---------+------------
 English | AG      |        100
 English | AXA     |        100
 English | BDS     |        100
 English | BVIR    |        100
 Spanish | C       |        100
 German  | D       |        100
 Spanish | DOM     |        100
 English | FALK    |        100
 French  | FGU     |        100
 German  | FL      |        100
(10 rows)

mondial2=# SELECT name, country, percentage FROM language WHERE percentage = 100 ORDER BY percentage DESC NULLS LAS
    name    | country | percentage
------------+---------+------------
 English    | AG      |        100
 English    | AXA     |        100
 English    | BDS     |        100
 English    | BVIR    |        100
 Spanish    | C       |        100
 German     | D       |        100
 Spanish    | DOM     |        100
 English    | FALK    |        100
 French     | FGU     |        100
 German     | FL      |        100
 English    | GBM     |        100
 English    | HELX    |        100
 Icelandic  | IS      |        100
 Japanese   | J       |        100
 English    | KN      |        100
 English    | MNTS    |        100
 Portuguese | MOC     |        100
 Burmese    | MYA     |        100
 Nepali     | NEP     |        100
 Dutch      | NL      |        100
 Korean     | NOK     |        100
 Portuguese | P       |        100
 Pitkern    | PITC    |        100
 Polish     | PL      |        100
 Russian    | R       |        100
 Spanish    | RA      |        100
 Spanish    | RCH     |        100
 French     | RG      |        100
 Italian    | RSM     |        100
 Arabic     | SA      |        100
 French     | SPMI    |        100
 Portuguese | STP     |        100
 English    | TUCA    |        100
(33 rows)
```

Then we did the aggregate query of which languages appeared in the most countries.

```
mondial2=# SELECT country, count(country) FROM language GROUP BY country ORDER BY count(country) DESC LIMIT 10;
```

```
 country | count
---------+-------
 PK      |     8
 IR      |     7
 SF      |     7
 AUS     |     6
 NLSM    |     6
 GNB     |     6
 NZ      |     6
 BZ      |     6
 MC      |     6
 A       |     6
(10 rows)
```

We noted that the count remain the same whether we counted by 'name' or 'country' (or just about anything that existed in the table). This is because what really matters is the grouping by 'country'.

```
mondial2=# SELECT country, count(name) FROM language GROUP BY country ORDER BY count(country) DESC LIMIT 10;
 country | count
---------+-------
 PK      |     8
 IR      |     7
 SF      |     7
 AUS     |     6
 NLSM    |     6
 GNB     |     6
 NZ      |     6
 BZ      |     6
 MC      |     6
 A       |     6
(10 rows)

mondial2=# SELECT country, count(percentage) FROM language GROUP BY country ORDER BY count(country) DESC LIMIT 10;
 country | count
---------+-------
 PK      |     8
 IR      |     7
 SF      |     7
 AUS     |     6
 NLSM    |     6
 GNB     |     6
 NZ      |     6
 BZ      |     6
 MC      |     6
 A       |     6
(10 rows)
```

We did another JOIN to get country names.

```
SELECT country.name, count(language.name) FROM language JOIN country ON country.code = language.country
mondial2-# GROUP BY country.name ORDER BY count(language.name) DESC LIMIT 10;
     name      | count
---------------+-------
 Pakistan      |     8
 Finland       |     7
 Iran          |     7
 Austria       |     6
 New Zealand   |     6
 Sint Maarten  |     6
 Australia     |     6
 Monaco        |     6
 Belize        |     6
 Guinea-Bissau |     6
```

Then we changed the grouping to get a COUNT of the most common languages.

```
SELECT name, count(name) FROM language GROUP BY name ORDER BY count(name) DESC LIMIT 10;
    name    | count
------------+-------
 English    |    35
 Spanish    |    26
 French     |    17
 Russian    |    14
 German     |    11
 Portuguese |    10
 Arabic     |     8
 Creole     |     6
 Serbian    |     6
 Turkish    |     6
(10 rows)
```

We added another column of aggregation--average population (this is still grouped by the language name).

```
SELECT name, count(name), avg(percentage) FROM language GROUP BY name ORDER BY count(name) DESC LIMIT 20;
    name    | count |         avg
------------+-------+---------------------
 English    |    35 | 55.9428571428571429
 Spanish    |    26 | 72.6423076923076923
 French     |    17 | 42.2647058823529412
 Russian    |    14 | 23.5857142857142857
 German     |    11 | 36.1363636363636364
 Portuguese |    10 | 43.9600000000000000
 Arabic     |     8 | 49.8125000000000000
 Turkish    |     6 | 18.8333333333333333
 Serbian    |     6 | 26.8833333333333333
 Creole     |     6 | 53.3833333333333333
 Albanian   |     5 | 44.2600000000000000
 Dutch      |     5 | 34.0400000000000000
 Italian    |     5 | 45.1200000000000000
 Roma       |     5 |  1.9400000000000000
 Hungarian  |     5 | 22.9800000000000000
 Greek      |     4 | 44.7000000000000000
 Croatian   |     3 | 35.2000000000000000
 Chinese    |     3 | 33.4000000000000000
 Ukrainian  |     3 | 24.0000000000000000
 Uzbek      |     3 | 32.3000000000000000
(20 rows)
```

We added HAVING to filter out all languages that appear in less than five countries. And we ordered by the average.

```
SELECT name, count(name), avg(percentage) FROM language GROUP BY name HAVING count(name) > 4ORDER BY avg(percentage
    name    | count |         avg
------------+-------+---------------------
 Spanish    |    26 | 72.6423076923076923
 English    |    35 | 55.9428571428571429
 Creole     |     6 | 53.3833333333333333
 Arabic     |     8 | 49.8125000000000000
 Italian    |     5 | 45.1200000000000000
 Albanian   |     5 | 44.2600000000000000
 Portuguese |    10 | 43.9600000000000000
 French     |    17 | 42.2647058823529412
 German     |    11 | 36.1363636363636364
 Dutch      |     5 | 34.0400000000000000
 Serbian    |     6 | 26.8833333333333333
 Russian    |    14 | 23.5857142857142857
 Hungarian  |     5 | 22.9800000000000000
 Turkish    |     6 | 18.8333333333333333
 Roma       |     5 |  1.9400000000000000
(15 rows)
```

Then we added WHERE to filter the table before the aggregation happens, and we get a much different result. We are now only counting languages when they are spoken by at least 21% of the population of the country. All of the other rows in the table are filtered out before the languages are grouped. (WHERE filters the table, HAVING filters the resulting aggregation.)

```
SELECT name, count(name), avg(percentage) FROM language WHERE percentage > 20 GROUP BY name HAVING count(name) > 4
  name   | count |         avg
---------+-------+---------------------
 English |    21 | 88.5428571428571429
 Spanish |    21 | 87.5666666666666667
 Arabic  |     5 | 79.2000000000000000
 German  |     5 | 76.9200000000000000
 French  |     9 | 73.5222222222222222
 Creole  |     5 | 62.4200000000000000
 Russian |     5 | 50.8000000000000000
(7 rows)
```

The thing to understand is that COUNT just counts number of rows. COUNT(distinct) counts the number of unique values.

```
mondial2=# SELECT count(distinct country) from language;
 count
-------
   130
```

```
(1 row)

mondial2=# SELECT count(distinct name) from language;
 count
-------
   108
(1 row)

mondial2=# select count(country) from language;
 count
-------
   294
(1 row)
```

So there are a total of 294 rows in the table. 130 unique countries, 108 unique languages.

So we started searching for specific languages using IN()

```
SELECT * FROM language WHERE name IN('English','Spanish') ORDER BY name;
 country |  name   | percentage
---------+---------+------------
 NMIS    | English |       10.8
 HONX    | English |        3.2
 PNG     | English |          1
 AXA     | English |        100
 AG      | English |        100
 BDS     | English |        100
 BZ      | English |         20
 BVIR    | English |        100
 CDN     | English |       58.8
 USA     | English |       82.1
 CAYM    | English |         95
 PA      | English |         14
 JA      | English |       63.5
 MNTS    | English |        100
 CUR     | English |        2.9
 NLSM    | English |       67.5
 KN      | English |        100
 TUCA    | English |        100
 AMSA    | English |        2.9
 AUS     | English |       78.5
 GUAM    | English |       38.3
 NZ      | English |       91.2
 SLB     | English |          1
 FALK    | English |        100
 NAM     | English |          7
 LB      | English |         20
 HELX    | English |        100
 L       | English |          1
 SF      | English |        0.3
 MC      | English |        8.5
 IRL     | English |         95
 GBJ     | English |       94.5
 M       | English |          6
 GBM     | English |        100
 GB      | English |         95
 E       | Spanish |         74
 PA      | Spanish |         84
mondial2=# SELECT * FROM language WHERE name IN('English','Spanish','Arabic') ORDER BY name;
 country |  name   | percentage
---------+---------+------------
 IL      | Arabic  |         23
 ET      | Arabic  |         99
 SF      | Arabic  |        0.3
 SA      | Arabic  |        100
 WEST    | Arabic  |         75
 IR      | Arabic  |          1
 AUS     | Arabic  |        1.2
 GAZA    | Arabic  |         99
 GB      | English |         95
 HONX    | English |        3.2
 PNG     | English |          1
 AXA     | English |        100
 AG      | English |        100
 BDS     | English |        100
 GUAM    | English |       38.3
 L       | English |          1
 SF      | English |        0.3
 MC      | English |        8.5
 IRL     | English |         95
 GBJ     | English |       94.5
 M       | English |          6
 GBM     | English |        100
 BZ      | English |         20
 BVIR    | English |        100
 CDN     | English |       58.8
 USA     | English |       82.1
 CAYM    | English |         95
 PA      | English |         14
 JA      | English |       63.5
 MNTS    | English |        100
 CUR     | English |        2.9
 NLSM    | English |       67.5
 KN      | English |        100
 TUCA    | English |        100
 AMSA    | English |        2.9
 AUS     | English |       78.5
 NZ      | English |       91.2
```

Then we did a JOIN to get GDP.

```
SELECT language.name, language.country, language.percentage, economy.gdp
mondial2-# FROM language JOIN economy ON language.country = economy.country
mondial2-# WHERE language.name IN('English','Spanish');
  name   | country | percentage |   gdp
---------+---------+------------+----------
 Spanish | AND     |         33 |     4800
 Spanish | E       |         74 |  1356000
 English | L       |          1 |    60540
 English | SF      |        0.3 |   259600
 English | MC      |        8.5 |     5748
 English | IRL     |         95 |   220900
 English | GBJ     |       94.5 |     5100
 English | M       |          6 |     9541
 English | GBM     |        100 |     4076
 English | GB      |         95 |  2490000
 English | HONX    |        3.2 |   272100
 English | PNG     |          1 |    16100
 English | AXA     |        100 |    175.4
 English | AG      |        100 |     1220
 English | BDS     |        100 |     4262
 Spanish | BZ      |         19 |     1637
 English | BZ      |         20 |     1637
 Spanish | GCA     |         60 |    53900
 Spanish | MEX     |         95 |  1327000
 English | BVIR    |        100 |     1095
 English | CDN     |       58.8 |  1825000
 English | USA     |       82.1 | 16720000
 Spanish | USA     |       10.7 | 16720000
 English | CAYM    |         95 |     2250
 Spanish | CAYM    |        3.2 |     2250
 Spanish | CR      |         99 |    48510
 Spanish | NIC     |       97.5 |    11260
 Spanish | PA      |         84 |    40620
 English | PA      |         14 |    40620
 Spanish | C       |        100 |    72300
 Spanish | DOM     |        100 |    59270
 Spanish | ES      |         99 |    24670
 Spanish | HCA     |         99 |    18880
 English | JA      |       63.5 |    14390
 English | MNTS    |        100 |       29
 English | CUR     |        2.9 |     5600
 Spanish | CUR     |          4 |     5600
```

Next we ordered by lowest GDP.

```
SELECT language.name, language.country, language.percentage, economy.gdp
mondial2-# FROM language JOIN economy ON language.country = economy.country
mondial2-# WHERE language.name IN('English','Spanish') ORDER BY economy.gdp;
  name   | country | percentage |   gdp
---------+---------+------------+----------
 English | HELX    |        100 |       18
 English | MNTS    |        100 |       29
 English | FALK    |        100 |    164.5
 English | AXA     |        100 |    175.4
 English | TUCA    |        100 |      216
 English | AMSA    |        2.9 |    462.2
 English | NMIS    |       10.8 |      733
 English | KN      |        100 |      767
 Spanish | NLSM    |       12.9 |    794.7
 English | NLSM    |       67.5 |    794.7
 English | BVIR    |        100 |     1095
 English | SLB     |          1 |     1099
 English | AG      |        100 |     1220
 Spanish | BZ      |         19 |     1637
 English | BZ      |         20 |     1637
 English | LB      |         20 |     1977
 English | CAYM    |         95 |     2250
 Spanish | CAYM    |        3.2 |     2250
 English | GBM     |        100 |     4076
 English | BDS     |        100 |     4262
 English | GUAM    |       38.3 |     4600
 Spanish | AND     |         33 |     4800
 English | GBJ     |       94.5 |     5100
 English | CUR     |        2.9 |     5600
 Spanish | CUR     |          4 |     5600
 English | MC      |        8.5 |     5748
 English | M       |          6 |     9541
 Spanish | NIC     |       97.5 |    11260
 English | NAM     |          7 |    12300
 English | JA      |       63.5 |    14390
 English | PNG     |          1 |    16100
 Spanish | GQ      |       67.6 |    17080
 Spanish | HCA     |         99 |    18880
 Spanish | ES      |         99 |    24670
 Spanish | PY      |         90 |    30560
 Spanish | BOL     |       60.7 |    30790
 English | PA      |         14 |    40620
```

Then we did a triple join to get the names of the countries. Note that JOIN is an extension of FROM.

```
SELECT language.name, language.country, language.percentage, economy.gdp, country.name
mondial2-# FROM language JOIN economy ON language.country = economy.country
mondial2-# JOIN country ON economy.country = country.code
mondial2-# WHERE language.name IN('English','Spanish') ORDER BY economy.gdp;
  name   | country | percentage |   gdp    |           name
---------+---------+------------+----------+--------------------------
 English | HELX    |        100 |       18 | Saint Helena
 English | MNTS    |        100 |       29 | Montserrat
 English | FALK    |        100 |    164.5 | Falkland Islands
 English | AXA     |        100 |    175.4 | Anguilla
 English | TUCA    |        100 |      216 | Turks and Caicos Islands
 English | AMSA    |        2.9 |    462.2 | American Samoa
 English | NMIS    |       10.8 |      733 | Northern Mariana Islands
 English | KN      |        100 |      767 | Saint Kitts and Nevis
 Spanish | NLSM    |       12.9 |    794.7 | Sint Maarten
 English | NLSM    |       67.5 |    794.7 | Sint Maarten
 English | BVIR    |        100 |     1095 | British Virgin Islands
 English | SLB     |          1 |     1099 | Solomon Islands
 English | AG      |        100 |     1220 | Antigua and Barbuda
 Spanish | BZ      |         19 |     1637 | Belize
 English | BZ      |         20 |     1637 | Belize
 English | LB      |         20 |     1977 | Liberia
 English | CAYM    |         95 |     2250 | Cayman Islands
 Spanish | CAYM    |        3.2 |     2250 | Cayman Islands
 English | GBM     |        100 |     4076 | Isle of Man
 English | BDS     |        100 |     4262 | Barbados
 English | GUAM    |       38.3 |     4600 | Guam
 Spanish | AND     |         33 |     4800 | Andorra
 English | GBJ     |       94.5 |     5100 | Jersey
 English | CUR     |        2.9 |     5600 | Curacao
 Spanish | CUR     |          4 |     5600 | Curacao
 English | MC      |        8.5 |     5748 | Monaco
 English | M       |          6 |     9541 | Malta
 Spanish | NIC     |       97.5 |    11260 | Nicaragua
 English | NAM     |          7 |    12300 | Namibia
 English | JA      |       63.5 |    14390 | Jamaica
 English | PNG     |          1 |    16100 | Papua New Guinea
 Spanish | GQ      |       67.6 |    17080 | Equatorial Guinea
 Spanish | HCA     |         99 |    18880 | Honduras
 Spanish | ES      |         99 |    24670 | El Salvador
 Spanish | PY      |         90 |    30560 | Paraguay
 Spanish | BOL     |       60.7 |    30790 | Bolivia
 English | PA      |         14 |    40620 | Panama
```

Here's a version with aliased names for the columns.

```
SELECT language.name AS lang, country.name AS cn, language.percentage AS lp, economy.gdp AS eg
mondial2-# FROM language JOIN economy ON language.country = economy.country
mondial2-# JOIN country ON economy.country = country.code
mondial2-# WHERE language.name IN('English','Spanish') ORDER BY economy.gdp;
  lang   |            cn            |  lp  |    eg
---------+--------------------------+------+----------
 English | Saint Helena             |  100 |       18
 English | Montserrat               |  100 |       29
 English | Falkland Islands         |  100 |    164.5
 English | Anguilla                 |  100 |    175.4
 English | Turks and Caicos Islands |  100 |      216
 English | American Samoa           |  2.9 |    462.2
 English | Northern Mariana Islands | 10.8 |      733
 English | Saint Kitts and Nevis    |  100 |      767
 Spanish | Sint Maarten             | 12.9 |    794.7
 English | Sint Maarten             | 67.5 |    794.7
 English | British Virgin Islands   |  100 |     1095
 English | Solomon Islands          |    1 |     1099
 English | Antigua and Barbuda      |  100 |     1220
 Spanish | Belize                   |   19 |     1637
 English | Belize                   |   20 |     1637
 English | Liberia                  |   20 |     1977
 English | Cayman Islands           |   95 |     2250
 Spanish | Cayman Islands           |  3.2 |     2250
 English | Isle of Man              |  100 |     4076
 English | Barbados                 |  100 |     4262
 English | Guam                     | 38.3 |     4600
 Spanish | Andorra                  |   33 |     4800
 English | Jersey                   | 94.5 |     5100
 English | Curacao                  |  2.9 |     5600
 Spanish | Curacao                  |    4 |     5600
 English | Monaco                   |  8.5 |     5748
 English | Malta                    |    6 |     9541
 Spanish | Nicaragua                | 97.5 |    11260
 English | Namibia                  |    7 |    12300
 English | Jamaica                  | 63.5 |    14390
 English | Papua New Guinea         |    1 |    16100
 Spanish | Equatorial Guinea        | 67.6 |    17080
 Spanish | Honduras                 |   99 |    18880
 Spanish | El Salvador              |   99 |    24670
 Spanish | Paraguay                 |   90 |    30560
 Spanish | Bolivia                  | 60.7 |    30790
 English | Panama                   |   14 |    40620
```

I surrounded the above query with ( ) and used the WITH command to turn it into a subtable/subquery. And then I aggregated it. This could have been done without the subquery, but it's a simple demonstration of how one works.

```
WITH langgdp as (
 SELECT language.name AS lang, country.name AS cn, language.percentage AS lp, economy.gdp AS eg
 FROM language JOIN economy ON language.country = economy.country
 JOIN country ON economy.country = country.code
WHERE language.name IN('English','Spanish') ORDER BY economy.gdp)
SELECT langgdp.lang, sum(langgdp.eg) FROM langgdp GROUP BY langgdp.lang;

  lang   |    sum
---------+------------
 Spanish | 21687741.7
 English | 23652214.8
(2 rows)
```

We dove into river.

```
\d river
                           Table "public.river"
      Column      |         Type          | Collation | Nullable | Default
------------------+-----------------------+-----------+----------+---------
 name             | character varying(50) |           | not null |
 river            | character varying(50) |           |          |
 lake             | character varying(50) |           |          |
 sea              | character varying(50) |           |          |
 length           | numeric               |           |          |
 area             | numeric               |           |          |
 source           | geocoord              |           |          |
 mountains        | character varying(50) |           |          |
 sourceelevation  | numeric               |           |          |
 estuary          | geocoord              |           |          |
 estuaryelevation | numeric               |           |          |
Indexes:
    "riverkey" PRIMARY KEY, btree (name)
Check constraints:
    "estcoord" CHECK ((estuary).latitude >= '-90'::integer::numeric AND (estuary).latitude <= 90::numeric AND (estu
    "riverarea" CHECK (area >= 0::numeric)
    "riverlength" CHECK (length >= 0::numeric)
    "rivflowsinto" CHECK (river IS NULL AND lake IS NULL OR river IS NULL AND sea IS NULL OR lake IS NULL AND sea I
    "sourcecoord" CHECK ((source).latitude >= '-90'::integer::numeric AND (source).latitude <= 90::numeric AND (sou
```

And looked for the longest rivers.

```
SELECT name, length FROM river ORDER BY length DESC NULLS LAST;
           name            | length
---------------------------+--------
 Yangtze                   |   6380
 Hwangho                   |   4845
 Lena                      |   4400
 Zaire                     |   4374
 Mekong                    |   4350
 Irtysch                   |   4248
 Niger                     |   4184
 Missouri                  |   4130
 Jenissej                  |   4092
 Amazonas                  |   3778
 Mississippi               |   3778
 Ob                        |   3650
 Volga                     |   3531
 Jurua                     |   3283
 Tarim-Yarkend             |   3260
 Purus                     |   3210
 Yukon River               |   3185
 Indus                     |   3180
 Nile                      |   3090
 Rio Grande del Norte      |   3034
 Saluen                    |   2980
 Brahmaputra               |   2896
 Rio Negro                 |   2866
 Volta                     |   2850
 Donau                     |   2845
 Angara                    |   2830
 Rio Sao Francisco         |   2830
 Amur                      |   2824
 Japura                    |   2816
 Darling River             |   2739
 Euphrat                   |   2736
 Parana                    |   2640
 Ganges                    |   2620
 Zambezi                   |   2574
 Paraguay                  |   2549
 Kolyma                    |   2513
 Tocantins                 |   2450
 Ischim                    |   2450
```

But where are these rivers? We have to look in geo_river to find out.

```
geo_river
                     Table "public.geo_river"
  Column  |         Type          | Collation | Nullable | Default
----------+-----------------------+-----------+----------+---------
 river    | character varying(50) |           | not null |
 country  | character varying(4)  |           | not null |
 province | character varying(50) |           | not null |
Indexes:
    "griverkey" PRIMARY KEY, btree (province, country, river)
```

We counted up the rivers that appear the most in the table geo_river. But we need to understand that each river appears numerous times, not by country but by province. The Donau goes through the most provinces.

```
SELECT river, count(river) FROM geo_river GROUP BY river ORDER BY count(river) DESC;
          river           | count
--------------------------+-------
 Donau                    |    33
 Niger                    |    20
 Rhein                    |    18
 Euphrat                  |    17
 Tigris                   |    16
 Zaire                    |    14
 Zambezi                  |    13
 Volga                    |    13
 Dnepr                    |    12
 Elbe                     |    12
 Rio Magdalena            |    11
 Yangtze                  |    11
 Parana                   |    11
 Theiss                   |    10
 Mississippi              |    10
 Ganges                   |    10
 Oder                     |    10
 Orinoco                  |     9
 Senegal                  |     9
 Irawaddy                 |     8
 Maas                     |     8
 Kasai                    |     8
 Sereth                   |     8
 Mekong                   |     8
 Aras                     |     8
 Cuango                   |     8
 Weichsel                 |     8
 Dnister                  |     8
 Drau                     |     8
 Brahmaputra              |     7
 Guadiana                 |     7
 Pruth                    |     7
 Volta                    |     7
 Blue Nile                |     7
 Bandama                  |     7
 Oka                      |     7
 March                    |     7
```

To find out which rivers go through the most countries we had to use COUNT(DISTINCT) which counts only unique countries. So it filters out all the repetitions of countries with multiple provinces. Note, that I didn't change the ORDER BY so it's still in the order of most provinces.

```
SELECT river, count(distinct country) FROM geo_river GROUP BY river ORDER BY count(river) DESC;
          river           | count
--------------------------+-------
 Donau                    |    10
 Niger                    |     4
 Rhein                    |     6
 Euphrat                  |     3
 Tigris                   |     3
 Zaire                    |     2
 Volga                    |     1
 Zambezi                  |     6
 Dnepr                    |     3
 Elbe                     |     2
 Rio Magdalena            |     1
 Parana                   |     3
 Yangtze                  |     1
 Theiss                   |     3
 Ganges                   |     2
 Mississippi              |     1
 Oder                     |     3
 Orinoco                  |     2
 Senegal                  |     4
 Maas                     |     3
 Dnister                  |     2
 Cuango                   |     2
 Weichsel                 |     1
 Kasai                    |     2
 Mekong                   |     6
 Drau                     |     5
 Aras                     |     4
 Sereth                   |     2
 Irawaddy                 |     2
 Brahmaputra               |     3
 Guadiana                  |     2
 Volta                     |     2
 ComoÃ©                     |     2
 Hwangho                   |     1
 Rio Grande del Norte      |     2
 Pruth                     |     3
 Blue Nile                 |     2
```

I added another column.

```
SELECT river, count(distinct country), count(province) FROM geo_river GROUP BY river ORDER BY count(river) DESC;
          river            | count | count
---------------------------+-------+-------
 Donau                     |    10 |    33
 Niger                     |     4 |    20
 Rhein                     |     6 |    18
 Euphrat                   |     3 |    17
 Tigris                    |     3 |    16
 Zaire                     |     2 |    14
 Volga                     |     1 |    13
 Zambezi                   |     6 |    13
 Dnepr                     |     3 |    12
 Elbe                      |     2 |    12
 Rio Magdalena             |     1 |    11
 Parana                    |     3 |    11
 Yangtze                   |     1 |    11
 Theiss                    |     3 |    10
 Ganges                    |     2 |    10
 Mississippi               |     1 |    10
 Oder                      |     3 |    10
 Orinoco                   |     2 |     9
 Senegal                   |     4 |     9
 Maas                      |     3 |     8
 Dnister                   |     2 |     8
 Cuango                    |     2 |     8
 Weichsel                  |     1 |     8
 Kasai                     |     2 |     8
 Mekong                    |     6 |     8
 Drau                      |     5 |     8
 Aras                      |     4 |     8
 Sereth                    |     2 |     8
 Irawaddy                  |     2 |     8
 Brahmaputra               |     3 |     7
 Guadiana                  |     2 |     7
 Volta                     |     2 |     7
 ComoÃ©                     |     2 |     7
 Hwangho                   |     1 |     7
 Rio Grande del Norte      |     2 |     7
 Pruth                     |     3 |     7
 Blue Nile                 |     2 |     7
```

Then I jumped in to mountain. To demonstrate SELECT DISTINCT which filters for unique values ACROSS columns. I want to find only one entry for each set of a mountain in the country(s) it is in. Why? Because the question was, "what country has the most mountains?" We can't just count up the mountains here because there are duplicates based on provinces. We can't just do unique mountains here because some mountains crossover in to multiple countries. So we want to reduce our table so there is one row for every unique mountain and country pair.

```
select distinct mountain, country from geo_mountain;
           mountain            | country
-------------------------------+---------
 CabeÃ§o Gordo                  | P
 Muztagh Ata                   | CN
 Krenizyn                      | R
 Illampu                       | BOL
 Baru                          | PA
 Poco Mandasawu                | RI
 Khuvkhoitun                   | R
 Mt. Fito                      | WS
 Toba Caldera                  | RI
 Pico das Agulhas Negras       | BR
 Makalu                        | NEP
 Katla                         | IS
 Pico de las Nieves            | E
 Dirfi                         | GR
 Taftan                        | IR
 Mt. Ulawun                      | PNG
 Pramnos                         | GR
 El Pital                        | HCA
 Ngá»c Linh                      | VN
 Shaiyb al-Banat                 | ET
 Phu Xai Lai Leng                | LAO
 Ichinsky                        | R
 Pic la Selle                    | RH
 Granite Peak                    | USA
 Popomanaseu                     | SLB
 Kasbek                          | R
 Anamudi                         | IND
 BazardÃ¼zÃ¼                      | R
 Pik Sedova                      | R
 Moldoveanu                      | RO
 Doddabetta                      | IND
 San Jacinto Peak                | USA
 Ishizuchi-San                   | J
 Alam Kuh                        | IR
 Pico                            | P
 Gasherbrum I                    | CN
 Lhotse                          | NEP
```

To demonstrate the problem I'm trying to solve. I did the following query. This counts the number of countries and the number of provinces for each mountain. Since the counts are quite different, and we are interested in countries, we want to get rid of all the duplicate rows. For example, be want to have 2 rows for Mt. Hermon, because it is in two countries, and get rid of the extra row for the country in which it is in two provinces. (!!!!)

```
select mountain, count(distinct country), count(distinct province)
mondial2-# from geo_mountain group by mountain order by count(distinct province) DESC;
            mountain             | count | count
---------------------------------+-------+-------
 Moldoveanu                      |     1 |     3
 Popocatepetl                    |     1 |     3
 Haku-San                        |     1 |     3
 Hiru Erregeen Mahaia            |     2 |     3
 Monte Rosa                      |     2 |     3
 Pik Manas                       |     3 |     3
 Zapaleri                        |     3 |     3
 Mt. Nimba                       |     3 |     3
 Mt. Hermon                      |     2 |     3
 Pik Chan-Tengri                 |     3 |     3
 Mousa Ali                       |     3 |     3
 Cerro Tristeza                  |     1 |     3
 K2                              |     2 |     2
 Llullaillaco                    |     2 |     2
 Mt. Everest                     |     2 |     2
 Mt. Fairweather                 |     2 |     2
 Ojos del Salado                 |     2 |     2
 OllagÃ¼e                         |     2 |     2
 Saramati                        |     2 |     2
 Olymp                           |     1 |     2
 Hochgolling                     |     1 |     2
 Schchara                        |     2 |     2
 Schneekoppe                     |     2 |     2
 Ararat                          |     1 |     2
 Kangchendzonga                  |     2 |     2
 Serra Dolcedorme                |     1 |     2
 Hotaka-Dake                     |     1 |     2
 Kanlaon                         |     1 |     2
 Maipo                           |     2 |     2
 Howerla                         |     1 |     2
 Makalu                          |     2 |     2
 Geladaindong                    |     1 |     2
 Phu Xai Lai Leng                |     2 |     2
 Shikengkong                     |     1 |     2
 Karisimbi                       |     2 |     2
 Bukit Raya                      |     1 |     2
 Kasbek                          |     2 |     2
```

To further clarify this issue, see how the first query--DISTINCT gives us only one entry for Cerro Tristeza, whereas without DISTINCT, we get 3.

```
mondial2=# select distinct mountain, country from geo_mountain ORDER by mountain;
            mountain             | country
---------------------------------+---------
```

```
 Aconcagua                      | RA
 Aenos                          | GR
 Agung                          | RI
 Alam Kuh                       | IR
 Alpamayo                       | PE
 Alto Toroni                    | BOL
 Alto Toroni                    | RCH
 Altun Shan Peak                | CN
 Ampato                         | PE
 Anamudi                        | IND
 Andringitra                    | RM
 Annapurna                      | NEP
 Aragaz                         | ARM
 Ararat                         | TR
 Arma Konda                     | IND
 Asahi-Dake                     | J
 Aso Rock                       | WAN
 Asralt Khairkhan               | MNG
 Athos                          | GR
 Attavyros                      | GR
 Ausangate                      | PE
 Ayrybaba                       | UZB
 Ayrybaba                       | TM
 Banahao                        | RP
 Barbeau Peak                   | CDN
 Barre des Ecrins               | F
 Baru                           | PA
 Batura Sar                     | PK
 BazardÃ¼zÃ¼                       | R
 BazardÃ¼zÃ¼                       | AZ
 Ben Nevis                      | GB
 Besar                          | RI
 Binaiya                        | RI
 Birch Mountain                 | CDN
 Bjelucha                       | R
 Bjelucha                       | KAZ
 Blue Mountain Peak             | JA
 Bobotov Kuk                    | MNE
 Borah Peak                     | USA
 Botew                          | BG
 Boundary Peak                  | USA
 Brandberg                      | NAM
 Broad Peak                     | CN
 Broad Peak                     | PK
 Brocken                        | D
 Bukadaban Feng                 | CN
 Bukit Batubrok                 | RI
 Bukit Raya                     | RI
 Buyu Balease                   | RI
 Buyu Lumut                     | RI
 CabeÃ§o Gordo                   | P
 Callaqui                       | RCH
 Carrauntoohil                  | IRL
 Cathkin Peak                   | RSA
 Cayambe                        | EC
 Cerro Chirripo                 | CR
 Cerro Fitzroy                  | RCH
 Cerro Fitzroy                  | RA
 Cerro Las Minas                | ES
 Cerro Las Minas                | HCA
 Cerro Mohinora                 | MEX
 Cerro San Rafael               | MEX
 Cerro Torre                    | RCH
 Cerro Torre                    | RA
 Cerro Tristeza                 | YV
 Cerro de Punta                 | PR
mondial2=# select mountain, country from geo_mountain ORDER by mountain;
            mountain            | country
--------------------------------+---------
 Aconcagua                      | RA
 Aenos                          | GR
 Agung                          | RI
 Alam Kuh                       | IR
 Alpamayo                       | PE
 Alto Toroni                    | RCH
 Alto Toroni                    | BOL
 Altun Shan Peak                | CN
 Ampato                         | PE
 Anamudi                         | IND
 Andringitra                     | RM
 Annapurna                       | NEP
 Aragaz                          | ARM
 Ararat                          | TR
 Ararat                          | TR
 Arma Konda                      | IND
 Asahi-Dake                      | J
 Aso Rock                        | WAN
 Asralt Khairkhan                | MNG
 Athos                           | GR
 Attavyros                       | GR
 Ausangate                       | PE
 Ayrybaba                        | TM
 Ayrybaba                        | UZB
 Banahao                         | RP
 Barbeau Peak                    | CDN
 Barre des Ecrins                | F
 Baru                            | PA
 Batura Sar                      | PK
 BazardÃ¼zÃ¼                       | R
 BazardÃ¼zÃ¼                       | AZ
 Ben Nevis                       | GB
 Besar                           | RI
 Binaiya                         | RI
 Birch Mountain                  | CDN
 Bjelucha                        | R
 Bjelucha                        | KAZ
 Blue Mountain Peak              | JA
 Bobotov Kuk                     | MNE
 Borah Peak                      | USA
 Botew                           | BG
 Boundary Peak                   | USA
 Boundary Peak                   | USA
 Brandberg                       | NAM
 Broad Peak                      | CN
 Broad Peak                      | PK
 Brocken                         | D
 Bukadaban Feng                  | CN
 Bukadaban Feng                  | CN
 Bukit Batubrok                  | RI
 Bukit Raya                      | RI
 Bukit Raya                      | RI
 Buyu Balease                    | RI
 Buyu Lumut                      | RI
 CabeÃ§o Gordo                    | P
 Callaqui                        | RCH
 Carrauntoohil                   | IRL
 Cathkin Peak                    | RSA
 Cayambe                         | EC
 Cerro Chirripo                  | CR
 Cerro Chirripo                  | CR
 Cerro Fitzroy                   | RA
 Cerro Fitzroy                   | RCH
 Cerro Las Minas                 | ES
 Cerro Las Minas                 | HCA
 Cerro Mohinora                  | MEX
 Cerro San Rafael                | MEX
 Cerro Torre                     | RA
 Cerro Torre                     | RCH
 Cerro Tristeza                  | YV
 Cerro Tristeza                  | YV
 Cerro Tristeza                  | YV
```

Finally--we can answer the question. We take that distinct table, and we turn into a subquery. And then we aggregate to get the country with the most mountains.

```
with countryMountain as (select distinct mountain, country from geo_mountain ORDER by mountain)
SELECT country.name, count(country.name) from countryMountain
JOIN country ON country.code = countryMountain.country
GROUP BY country.name ORDER BY count(country.name) DESC;
               name               | count
----------------------------------+-------
 China                            |    49
 Indonesia                        |    49
 United States                    |    37
 Russia                           |    29
 Chile                             |    19
 Philippines                       |    18
 Canada                            |    17
 Italy                             |    17
 Spain                             |    16
 Argentina                         |    15
 Greece                            |    14
 India                             |    12
 Australia                         |    10
 Pakistan                          |    10
 Papua New Guinea                  |     9
 France                            |     9
 Japan                             |     9
 Turkey                            |     8
 Nepal                             |     8
 Bolivia                           |     8
 Mexico                            |     8
 Peru                              |     8
 Portugal                          |     7
 Iran                              |     7
 Tajikistan                        |     7
 Brazil                            |     6
 Kyrgyzstan                        |     6
 Switzerland                       |     5
 Myanmar                           |     5
 New Zealand                       |     5
 Mongolia                          |     5
 Iceland                           |     4
 United Kingdom                    |     4
 Germany                           |     4
 Colombia                          |     4
 Vietnam                           |     4
 Kazakhstan                        |     4
```

Yay!!!