Drugi sprawdzian z SQL 20.04.2021

Dla każdego z poniższych zadań napisz odpowiednie polecenia SQL w zadaniu w SKOSie. Oczekujemy rozwiązania w postaci pliku zawierającego TREŚCI poleceń SQL, a nie znalezionej odpowiedzi. Nie będą sprawdzane jakiekolwiek zapytania niepoprawne składniowo, sprawdź swoje rozwiązanie używając `\i plik.sql` ! Plik możesz wysyłać wielokrotnie, sprawdzana będzie wyłącznie najnowsza wersja.

Wczytaj do swojej bazy danych plik `mondial-inputs.sql` znajdujący się w archiwum `https://skos.ii.uni.wroc.pl/mod/resource/view.php?id=23025` na SKOSIE.

Zachęcam do korzystania przede wszystkim z dokumentacji PostgreSQL: `https://www.postgresql.org/docs/11/index.html`.

Format nazwy pliku z rozwiązaniem: `grupa-imie-nazwisko.sql`, gdzie `grupa` to inicjały prowadzącego Twoją grupę: (`pwi/ppo/mpy/mabi/pga/plg`), np. `pwi-Jan-Kowalski.sql`. Wymagany format pliku z rozwiązaniem (tu też podaj swoje imię, nazwisko i grupę):

```
-- Imię Nazwisko, grupa np. Jan Kowalski, pwi
-- Zadanie 1
<zapytanie>

-- Zadanie 2
<zapytanie>
```

Zadanie 1 (2 pkt.) Dla każdego kraju wypisz jego nazwę oraz liczbę wysp położonych na graniczących z nim zbiornikach wodnych (`sea`). Użyj tabel: `country`, `islandin`, `geo_sea`). Wyniki posortuj według tej liczby malejąco, w drugiej kolejności według nazwy kraju rosnąco.

Rozwiązanie Aby wypisać wszystkie kraje, w tym kraje z 0 pasujących wysp, należało użyć łączenia zewnętrznego. Wiele osób zapomniało też o słowie kluczowym `DISTINCT`, aby nie liczyć wielokrotnie wysp, które zachodzą na kilka zbiorników wodnych.

```
SELECT Name, COUNT(DISTINCT island)
  FROM Country
    LEFT JOIN geo_Sea ON Code = Country
    LEFT JOIN islandIn USING (sea)
  GROUP BY Code
  ORDER BY 2 DESC, 1;
```

Zadanie 2 (2 pkt.) Znajdź kraje, o których według bazy wiadomo, że mieszkają w nich przedstawicielki/-le co najmniej 10 grup etnicznych, w tym Polacy. Przyjmujemy, że grupę etniczną jednoznacznie identyfikuje wartość atrybutu `name` relacji `ethnicgroup`. W wyniku podaj nazwę kraju oraz procent populacji tego kraju, który stanowią Polacy.

Rozwiązanie

```
SELECT Country.Name, Poles.Percentage
  FROM Country JOIN EthnicGroup AS Poles ON Code = Country
    JOIN EthnicGroup AS AllGroups USING (Country)
  WHERE Poles.Name = 'Polish'
  GROUP BY Code, Poles.Percentage
  HAVING COUNT(AllGroups.Name) >= 10;
```

Zadanie 3 (2 pkt.) Wypisz wszystkie kraje, do których da się dotrzeć z Polski przekraczając (być może wielokrotnie) wyłącznie granice opisane w tabeli `borders`. Załóżmy, że pomiędzy dowolnymi dwoma punktami każdego kraju możemy się poruszać bez przekraczania granic (np. samolotem). Dla każdego z takich krajów wypisz pełną krotkę jego atrybutów z tabeli `country`. Wyniki posortuj alfabetycznie według nazwy kraju. **Uwaga**: uwzględnij, że relacja opisana w `borders` nie jest symetryczna, ale naturalnie rozumiana relacja graniczenia już tak.

Rozwiązanie Będziemy generować zbiór państw `foo`, do których można dotrzeć z Polski. Na początku `foo` będzie się składać tylko z Polski:

```
WITH RECURSIVE foo AS (
  SELECT 'PL'::VARCHAR(4) AS Code
  UNION
  ...
SELECT * FROM Country NATURAL JOIN foo;
```

W miejscu "..." należy określić, jak wygląda jedna iteracja dodawania elementów do naszego zbioru `foo`. W miejsce "..." należy więc wstawić zapytanie, które zwróci nam państwa sąsiadujące z tymi, które już są w naszym zbiorze `foo` (pamiętajmy, że relacja `borders` nie jest symetryczna w naszej bazie):

```
  SELECT Code2 AS Code
    FROM (
      SELECT Country1 AS Code, Country2 AS Code2 FROM Borders
      UNION
      SELECT Country2 AS Code, Country1 AS Code2 FROM Borders
    ) bar
      NATURAL JOIN foo
```

Zatem pełne rozwiązanie wygląda tak:

```
WITH RECURSIVE foo AS (
  SELECT 'PL'::VARCHAR(4) AS Code
  UNION
  SELECT Code2 AS Code
```

```
    FROM (
      SELECT Country1 AS Code, Country2 AS Code2 FROM Borders
      UNION
      SELECT Country2 AS Code, Country1 AS Code2 FROM Borders
    ) bar
      NATURAL JOIN foo
)
SELECT * FROM Country NATURAL JOIN foo;
```

Zadanie 4 (2 pkt.) Znajdź kraje, w których co najmniej 75% populacji mieszka w miastach wymienionych w tabeli `city`. W wyniku podaj nazwę kraju oraz procent populacji mieszkający w miastach w postaci liczby całkowitej wyrażającej procent. Wyniki posortuj malejąco względem tej liczby. (Nie musisz brać pod uwagę historycznych danych o populacji z tabel `countrypops`, `citypops`.)

Rozwiązanie

```
SELECT Country.Name,
       FLOOR(100 * SUM(City.Population) / Country.Population) AS foo
  FROM Country JOIN City ON Country = Code
  GROUP BY Code
  HAVING FLOOR(100 * SUM(City.Population) / Country.Population) > 75
  ORDER BY 2 DESC;
```