

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ - ПРОЦЕССОВ УПРАВЛЕНИЯ**

Сердюков Дмитрий Русланович

Курсовая работа

**ЕВРОПЕЙСКИЕ СТРАНЫ
(EUROPEAN COUNTRIES)**

Направление 010302

Прикладная математика и информатика

Научный руководитель,
Старший преподаватель

Малинин К.А.

Санкт-Петербург
2017

Содержание

Глава 1. Описание и структура.....	2
Глава 2. Лёгкие запросы.....	4
Глава 3. Средние запросы.....	6
Глава 4. Сложные запросы.....	9

Все файлы располагаются на гитхабе по адресу:
github.com/shumoff/database-european_countries

Глава 1. Описание и структура.

База данных «European Countries» рассказывает о положении дел в Европе. В ней содержится вся основная информация по каждой из стран - от демографических показателей до политической ситуации.

В базе реализуются 7 объектов:

Countries - Список стран Европы с некоторыми основными характеристиками

Поля:

- id - идентификатор страны (primary key)
- name - название страны (unique)
- capital - столица страны (unique)
- population - население страны
- territory_sq_km - площадь страны
- region_id - идентификатор региона, к которому относится страна (foreign key)

Regions - Список регионов Европы (всего 4) с дополнительными данными

Поля:

- rid - идентификатор региона (primary key)
- name - название региона (unique)
- economical_leader - самая экономически развитая страна региона (foreign key) (unique)

Major International Associations - Основные международные объединения (такие как Совет Европы или УЕФА)

Поля:

- aid - идентификатор международного объединения (primary key)
- name - название объединения (unique)
- type - тип объединения (военное, спортивное, экономическое и т.д.)
- chairman - председатель/генеральный секретарь/президент
- headquarters_city - город, в котором находится штаб-квартира
- year_of_foundation - год основания объединения

Languages - Языки, на которых говорят в стране

Поля:

- lid - идентификатор языка (primary key)
- name - название языка (unique)

Religions - Религии, которые исповедуют в стране

Поля:

- rel_id - идентификатор религии (primary key)
- name - название религии (unique)

Rulers - Список правителей (не только текущие, могут быть добавлены и предыдущие, поэтому соотношение **1:m**)

Поля:

- ruler_id - идентификатор правителя (primary key)
- country_id - идентификатор страны (foreign key)
- name - имя правителя {unique}
- beginning of government - год начала правления по двум столбцам}
- end of government - год окончания правления

Political parties - Список главных партий Европы

Поля:

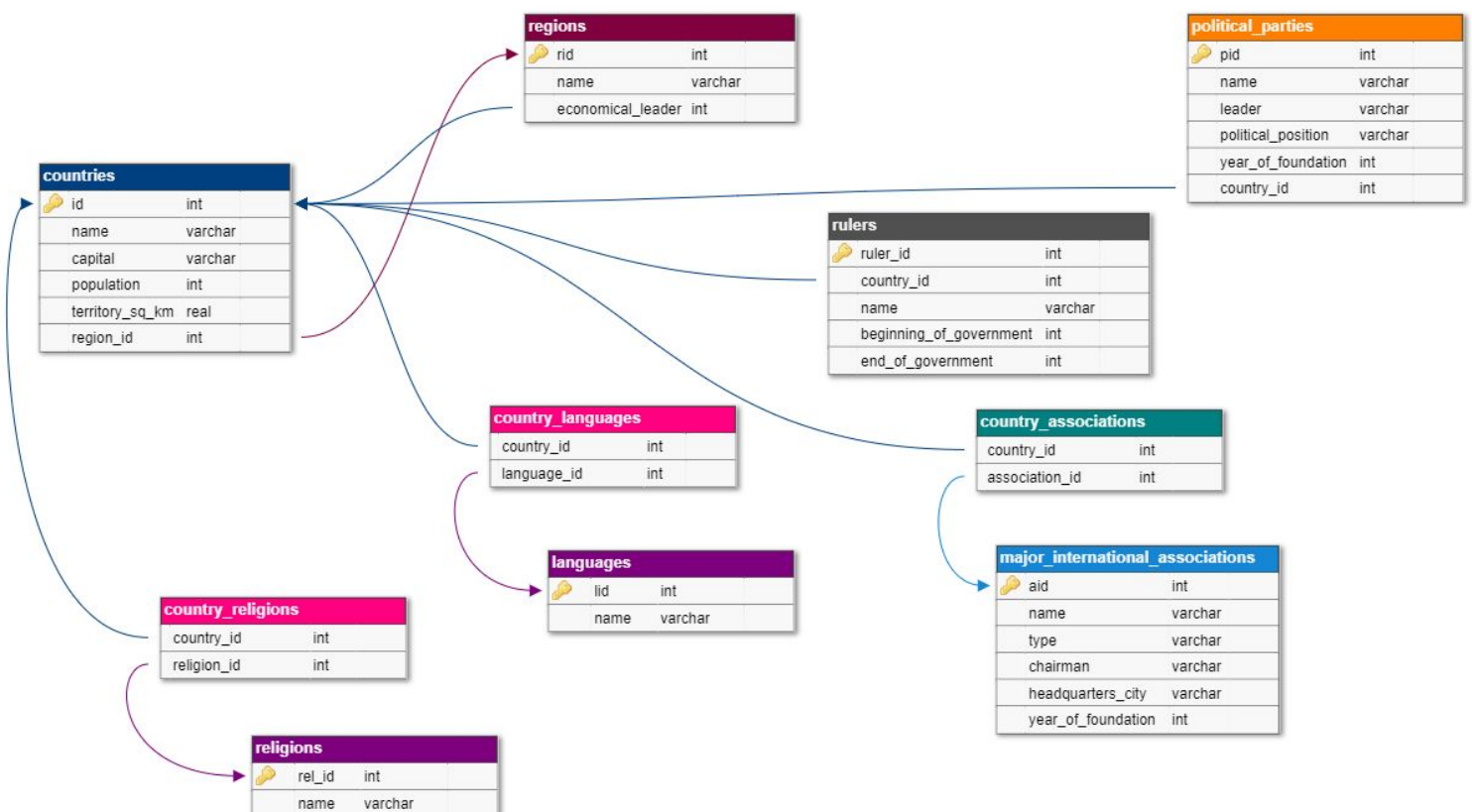
- pid - идентификатор партии (primary key)
- name - название партии {unique}
- leader - лидер партии по двум столбцам}
- political position - политический уклон (правые, левые и т.д.)
- year of foundation - год основания партии
- country_id - идентификатор страны (foreign key)

Задействуются отношения:

- countries **m:m** major_international_associations (country_associations)
- countries **m:m** languages (таблица country_languages)
- countries **m:m** religions (таблица country_religions)
- countries **1:m** political_parties
- countries **1:m** rulers
- countries **m:1** regions



dbdesigner.net



Глава 2. Лёгкие запросы.

1. Выводит все страны Европы, отсортированные по убыванию их площади.

```
SELECT id, name, territory_sq_km  
FROM countries  
ORDER BY territory_sq_km DESC;
```

Оптимизация:

Добавлен индекс `countries_territory_sq_km_idx` для сортировки стран по площади.

```
CREATE INDEX ON countries(territory_sq_km);
```

2. Выводит население выбранного региона(суммарное население всех стран в регионе).

```
SELECT sum(population)  
FROM countries  
WHERE region_id=1;
```

Оптимизация:

Добавлен индекс `countries_region_id_idx` для фильтрации стран по региону.

```
CREATE INDEX ON countries(region_id);
```

3. Выводит перечень типов международных объединений.

```
SELECT DISTINCT ON (type) type AS types_of_associations  
FROM major_international_associations;
```

Оптимизация:

Добавлен индекс `major_international_associations_type_idx` для фильтрации по уникальности типов международных объединений.

```
CREATE INDEX ON major_international_associations(type);
```

4. Выводит все политические партии, основанные в 20м веке и отсортированные по году основания.

```
SELECT *  
FROM political_parties  
WHERE year_of_foundation > 1899  
ORDER BY year_of_foundation;
```

Оптимизация:

Добавлен индекс `political_parties_year_of_foundation_idx` для фильтрации и сортировки по году основания партии.

```
CREATE INDEX ON political_parties(year_of_foundation);
```

Глава 3. Средние запросы.

1. Выводит партии, у которых лидер на данный момент является действующим правителем.

```
SELECT p.country_id, c.name AS country,  
p.pid, p.name AS party,  
r.ruler_id, p.leader AS ruler  
FROM political_parties p JOIN  
rulers r ON r.name=p.leader JOIN  
countries c ON r.country_id=c.id  
WHERE r.end_of_government is NULL;
```

Оптимизация:

Добавлены индексы `rulers_name_idx`, `rulers_country_id_idx`, `political_parties_leader_idx`; используется индекс `countries_pkey` для оптимизации объединения таблиц.

```
CREATE INDEX ON rulers(name text_pattern_ops);
```

```
CREATE INDEX ON rulers(country_id);
```

```
CREATE INDEX ON political_parties(leader text_pattern_ops);
```

Добавлен индекс `rulers_end_of_government` для фильтрации по году окончания правления.

```
CREATE INDEX ON rulers(end_of_government) WHERE(end_of_government is NULL);
```

2. Выводит все международные объединения, отсортированные по убыванию количества стран-участниц.

```
SELECT m.aid, m.name AS assosiation,  
count (*) AS amount_of_countries_in_association  
FROM major_international_associations m JOIN  
country_associations c ON m.aid=c.association_id  
GROUP BY m.aid  
ORDER BY amount_of_countries_in_association DESC;
```

Оптимизация:

Используются индексы `major_international_associations_pkey`,
`country_associations_pkey` для оптимизации объединения таблиц.

3. Выводит все регионы, отсортированные по убыванию площади.

```
SELECT r.rid, r.name AS region,  
sum(c.territory_sq_km) AS region_territory_sq_km  
FROM countries c JOIN regions r ON c.region_id=r.rid  
GROUP BY r.rid  
ORDER BY region_territory_sq_km DESC;
```

Оптимизация:

Добавлен индекс `countries_region_id_idx`, используется индекс `regions_pkey` для оптимизации объединения таблиц.

```
CREATE INDEX ON countries(region_id);
```


4. Выводит все регионы, отсортированные по убыванию населения.

```
SELECT r.rid, r.name AS region,  
sum(c.population) AS region_population  
FROM countries c JOIN regions r ON c.region_id=r.rid  
GROUP BY r.rid  
ORDER BY region_population DESC;
```

Оптимизация:

Используются индексы **countries_region_id_idx** и **regions_pkey** для оптимизации объединения таблиц.

Глава 4. Сложные запросы.

Следующие сложные запросы располагаются в логически обоснованном порядке (от глобальных структур к более локальным). Основными являются 1.3, 2.1, 3.2.

1.1 Выводит все страны Европы, отсортированные по убыванию плотности населения.

```
SELECT c1.id, c1.name AS country,  
c1.territory_sq_km,  
c1.population,  
round(  
    (SELECT (c1.population)  
    AS foo) /  
    (SELECT (c1.territory_sq_km)  
    AS foo))  
AS population_density  
FROM  
countries c1 JOIN  
countries c2 ON (c1.id=c2.id AND c1.region_id = 1)  
GROUP BY c1.id  
ORDER BY population_density DESC;
```

1.2 Выводит все регионы Европы, отсортированные по убыванию плотности населения.

```
SELECT r.rid, r.name AS region,  
sum(c.territory_sq_km) AS region_territory_sq_km,  
sum(c.population) AS region_population,  
round(  
    (SELECT AVG(region_population) FROM  
        (SELECT sum(c.population) AS region_population)  
        AS foo) /  
    (SELECT AVG(region_territory_sq_km) FROM  
        (SELECT sum(c.territory_sq_km) AS region_territory_sq_km)  
        AS foo))  
AS region_population_density  
FROM countries c JOIN  
regions r ON c.region_id=r.rid  
GROUP BY r.rid  
ORDER BY region_population_density DESC;
```

1.3 Выводит все страны в выбранном регионе, отсортированные по убыванию плотности населения.

```
SELECT c1.id, c1.name AS country,  
c1.territory_sq_km, c1.population,  
round(  
    (SELECT (c1.population)  
    AS foo) /  
    (SELECT (c1.territory_sq_km)  
    AS foo))  
AS population_density  
FROM countries c1 JOIN  
countries c2 ON (c1.id=c2.id AND c1.region_id = 1)  
GROUP BY c1.id  
ORDER BY population_density DESC;
```

2.1 Выводит все страны Европы, отсортированные по убыванию в стране количества партий с выбранной политической позицией.

```
SELECT c.name AS country,  
count(p.pid) AS number_of_parties  
FROM countries c LEFT JOIN political_parties p  
ON (c.id = p.country_id AND p.political_position = 'centre-right')  
GROUP BY c.name  
ORDER BY number_of_parties DESC;
```

3.1 Выводит самый распространённый язык (по количеству стран, в которых он представлен) в Европе.

```
SELECT l.name AS the_most_presented_language,  
count(*) AS amount_of_countries  
FROM languages l JOIN  
country_languages cl ON (l.lid = cl.language_id)  
GROUP BY l.lid  
      HAVING count(*) =  
      (SELECT MAX(amount_of_countries)  
      FROM (SELECT count(*) AS amount_of_countries  
            FROM languages l JOIN country_languages cl  
            ON (l.lid = cl.language_id) GROUP BY l.lid)  
      AS t)  
ORDER BY l.lid;
```

3.2 Выводит самый распространённый язык (по количеству стран, в которых он представлен) в выбранном регионе.

```
WITH t1 AS
  (SELECT l.id, l.name
   FROM languages l JOIN country_languages cl
   ON (l.id = cl.language_id) JOIN countries c
   ON (c.region_id = 4 AND cl.country_id = c.id))
SELECT t1.name AS the_most_presented_language,
count(*) AS amount_of_countries
FROM t1 GROUP BY t1.name
HAVING count(*) =
  (SELECT MAX(amount_of_countries)
   FROM (SELECT count(*) AS amount_of_countries
        FROM t1 GROUP BY t1.name)
   AS t2)
ORDER BY t1.name;
```