**Exercise 1**

**Show the biggest 3 countries on the least populous continent (except for Antarctica).**

**(Comment: please don’t hardcode ‘Oceania’.)**

select continent, name as country\_name, population,

sum(population) over (partition by continent) as continent\_pop

from country

where population > 0

order by continent\_pop,population Desc

limit 3

**Exercise 2**

**Show all countries with at least 3 official languages, order them by their surface area.**

select c.code, c.surfacearea, cc.off\_lang

from country as c

left join

(select countrycode, count(isofficial) as off\_lang

from countrylanguage

where isofficial = TRUE

group by countrycode) as cc

on c.code = cc.countrycode

where cc.off\_lang >= 3

order by c.surfacearea desc

**Exercise 3**

**For the 20 oldest countries (by indepyear), show the difference between life expectancy in each country and the continental average.**

select c1.name, c1.indepyear as independence, c1.lifeexpectancy- sq1.avglife as difference

from country as c1,

(select continent, avg(lifeexpectancy) as avglife

from country group by continent) sq1

where sq1.continent= c1.continent

order by c1.indepyear

limit 20

**Exercise 4**

**Show the 20 biggest cities in the United States along with their rank in the state and percent of the city population in a state.**

**(Comment: note that this database does not contain all municipalities. Restrict your population count to this dataset.)**

SELECT [ALL | DISTINCT] column1[,column2] FROM table1[,table2]

[WHERE "conditions"] [GROUP BY "column-list"]

[HAVING "conditions]

[ORDER BY "column-list" [ASC | DESC]

select ci.name, ci.population\*100/sq1.statepop

as state\_perc,

rank() over(partition by ci.district order by ci.population desc) as state\_rank

from city as ci, country as co,

(select district, sum(population) as statepop from city as ci

group by district) sq1

where ci.countrycode = co.code AND ci.countrycode='USA' AND

ci.district=sq1.district

order by ci.population desc

limit 20

**Exercise 5**

**For the 5 most populous countries, show the 3 most populous cites per country. Order cities by their population. with bigcity as (select name, population,**

rank() over(partition by countrycode order by population desc)

as rank from city)

select co.name, ci.name, ci.population,

rank() over(partition by ci.countrycode order by ci.population desc)

as city\_rank from city as ci, country as co, bigcity as bc,

(select name, population as countpop from country

order by countpop desc limit 5) bigco

where ci.countrycode = co.code and ci.name=bc.name and co.name= bigco.name

and bc.rank<=5

with co as (select name, code , rank() over(order by population desc)

from country rank),

ci as (select name, countrycode , rank() over(partition by countrycode order by population desc)

from city rank)

select co.name, ci.name,co.rank, ci.rank

from co

inner join ci

ON co.code=ci.countrycode

where ci.rank<=3 and co.rank <=5

order by co.rank, ci.rank

**Exercise 6**

**You run a travel agency and want to create a database for your business.**

**Right all data is being stored in a text file, which looks like that:**

*Picturesque Polish Ponds*

*1-5 May 2017*

*Guides: Andrew Brown (EN, MN, PL), Teresa Kowalska (EN, PL, RU)*

*Participants:*

*\* Hanna Edison <hanna.edison.137@gmail.com> (£750, paid)*

*\* Sue Clark <sue@sueclark.uk> (£750, unpaid, booking due: 31 Dec 2016)*

*\* Gordon Ryan <gr@halfbiolife.com> (£750, paid)*

*Marvellous Mongolian Mountains*

*14-30 June 2017*

*Guides: Rachel Diaz (ES, PR, RU, MN), Andrew Brown (EN, MN, PL)*

*Participants:*

*\* …*

**Create PostgreSQL tables that would be capable of storing this data. Present this data in a normal form (i.e. without multiple entries in the same field like ‘EN, MN, PL’, and without data duplication).**

drop table humans;

drop table marriages;

#creating a table containing names of people that are to be or have been married

create table humans (

id integer PRIMARY KEY,

forename text NOT NULL,

surname text NOT NULL,

birthday date,

sex varchar(1)

constraint chk\_sex CHECK(sex in ('m','f','M','F')));

#creating a table which will have information on conducted marriages

create table marriages (

id integer PRIMARY KEY,

partner\_1\_id integer,

partner\_2\_id integer,

marriage\_date date,

divorce\_date date,

constraint chk\_date CHECK (marriage\_date<divorce\_date),

foreign key (partner\_1\_id) REFERENCES humans(id),

foreign key (partner\_2\_id) REFERENCES humans(id));

create table languages (

lang\_id integer primary key,

language text not null)

create table guides (

guide\_id integer PRIMARY KEY,

first\_name text NOT NULL,

last\_name text NOT NULL,

employment\_date date,

lang1 integer,

lang2 integer,

lang3 integer,

lang4 integer,

lang5 integer,

foreign key (lang1) REFERENCES languages(lang\_id),

foreign key (lang2) REFERENCES languages(lang\_id),

foreign key (lang3) REFERENCES languages(lang\_id),

foreign key (lang4) REFERENCES languages(lang\_id),

foreign key (lang5) REFERENCES languages(lang\_id))

create table tours (

tour\_id integer PRIMARY KEY,

tour\_name text NOT NULL,

start\_date date,

end\_date date,

price integer not null,

guide\_id integer,

foreign key (guide\_id) REFERENCES guides(guide\_id))

create table customers (

cust\_id integer PRIMARY KEY,

first\_name text NOT NULL,

last\_name text NOT NULL,

email\_add text NOT NULL)

create table cust\_payments (

tour\_id integer primary key,

cust\_id integer,

amount\_paid integer,

outstanding\_amount integer,

booking\_date date,

foreign key (tour\_id) REFERENCES tours(tour\_id),

foreign key (cust\_id) REFERENCES customers(cust\_id))

drop table languages;

drop table guides cascade;

drop table customers cascade;

drop table cust\_payments cascade;

drop table tours cascade;

select \* from languages;

select \* from tours;

select \* from guides;

select \* from customers;

select \* from cust\_payments;