BT5110

Test 1

INSTRUCTIONS

1. This assessment **starts at 18:40**.
2. This assessment **ends at 20:30**.
3. The **maximum mark is 25.**
4. This is an **open book, open computer, and open Internet assessment**.
5. You are not allowed to communicate with anyone but members of the teaching team. Shall you use any external source of information make sure that you include a reference to the source in your answer (e.g. the URL).
6. Any student who is alleged to have committed or attempted to commit, or caused or attempted to cause any other person to commit any of the following offences: plagiarism, giving or receiving unauthorised assistance in academic work, or other forms of academic dishonesty, may be subject to disciplinary proceedings.
7. **Download this question paper and the template answer file**:
   - "`test1.pdf`",
   - "`answers.sql`",

   from the Luminus directory:

   "`Files > Tests > Test 1`".
8. **Download the following files**:
   - "`app.sql`",
   - "`appfunctionality.sql`",
   - "`available.sql`",
   - "`country.sql`",
   - "`functionality.sql`", and
   - "`store.sql`".

   from the Luminus directory:

   "`Files > Cases > Covid19`".
9. **Write your student number** in the corresponding section of the file "`answers.sql`".
10. **Write your answers** to the questions in the corresponding sections of the file "`answers.sql`".
11. **Within the 10 minutes following the end of the assessment, upload the file** "`answers.sql`" to the Luminus directory:

    "`Files > Tests > Test 1 > Submissions`".

1. The European Commission Joint Research Centre maintains a data set of mobile applications (apps) published across the world to fight the COVID-19 crisis. We consider a simplified subset of the data set stored in a relational schema comprising six self-describing tables.

   Read the SQL data definition language code in the SQL files provided and explore the provided instances of the tables to understand the application. Notice that the table country is denormalised.

   Prefer simple queries to nested queries, algebraic queries, and aggregate queries, in this order unless otherwise specified or strictly necessary. Do not use subqueries in the `FROM` clause unless otherwise specified or absolutely necessary. Do not create temporary tables, views, or stored functions and triggers unless otherwise specified or absolutely necessary. Your answers should be correct for the example database instance and for other instances of the same schema (for instance, new or updated apps, stores, functionalities, etc.). Your SQL code must run on PostgreSQL version 13 and above.

   (a) (3 points) (SQL) Print the names and ISO two letter codes of the different continents. The result should be similar to that in the table below in any order.

continent_name	continent_code
"Africa"	"AF"
"Europe"	"EU"
"South America"	"SA"
"Asia"	"AS"
"North America"	"NA"
"Oceania"	"OC"
"Antarctica"	"AN"

   (b) (3 points) (SQL) Find the contact tracing apps that are available in Europe and work for both iOS and Android operating systems. For each app, print the name of the app (rename the column as "`app`") and the name of the country (rename the column as "`country`") in which the app is available. The result should be similar to that in the table below in any order.

app	country
"COVID-19.eus"	"Spain, Kingdom of"
"Rakning C-19"	"Iceland, Republic of"
"Stopp Corona"	"Austria, Republic of"
"Corona-Warn-App"	"Germany, Federal Republic of"
"Immuni"	"Italy, Italian Republic"
"StopKorona!"	"Macedonia, The Former Yugoslav Republic of"
"SwissCovid"	"Switzerland, Swiss Confederation"
"ProteGO Safe"	"Poland, Republic of"
"Apturi Covid Latvia – SPKC"	"Latvia, Republic of"
"StopCovid France"	"France, French Republic"

   (c) (3 points) (SQL) Find the names of the countries that are spanning over several continents. Use aggregate functions. The result should be similar to that in the table below in any order.

name
"Armenia, Republic of"
"Azerbaijan, Republic of"
"Cyprus, Republic of"
"Georgia"
"Kazakhstan, Republic of"
"Russian Federation"
"Turkey, Republic of"
"United States Minor Outlying Islands"

   (d) (3 points) (SQL) Find the names of the countries that are spanning over several continents. This is the same query as (1.c) only do not use an aggregate query. The result should be similar to that in the table above in any order.

(e) (3 points) (SQL) Find the names of the apps available in countries in Oceania that work for all recorded operating systems. The result should be similar to that in the table below in any order. Make sure that your query produces the correct result in all possible database instances including when new apps with new operating systems are added (you may neither hardcode the operating systems as in Question 1.b nor the number of operating systems.) Do not use an aggregate query.

name
"TCC COVID Monitor"
"healthdirect"
"Coronavirus Australia"
"MyAus COVID-19"
"#BeatCovid19Now"
"Bupa Aged Care Connect"
"NZ COVID Tracer"
"COVIDSafe"

(f) (3 points) (SQL) Find the countries with the top 6 largest number of apps available (this is a dense ranking of the countries according to the number of apps). Print the names of the countries and the numbers of apps in descending order of the numbers of apps. The result should be similar to that in the table below. For full mark, make sure that your query produces the correct result for the top 7, 8 to 16 as well. Do not use "`RANK()`", "`DENSE_RANK()`", "`ROW_NUMBER()`", window functions and partitions (although they could be better choices in this case.) Use "`GROUP BY`", "`ORDER BY`", and "`LIMIT`".

name	count
"United States of America"	65
"India, Republic of"	52
"Mexico, United Mexican States"	21
"Brazil, Federative Republic of"	17
"Spain, Kingdom of"	16
"United Kingdom of Great Britain & Northern Ireland"	14

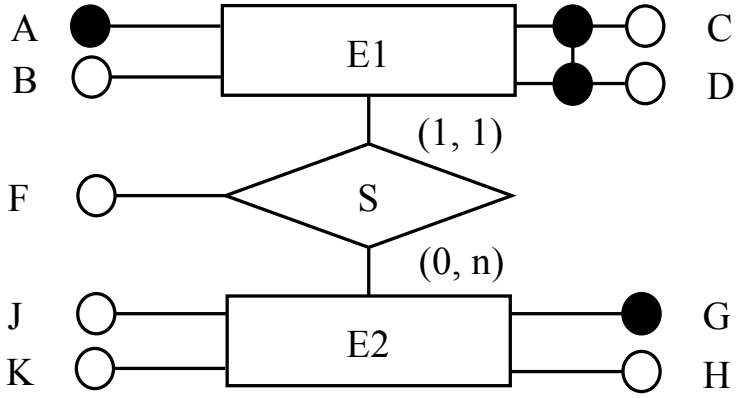

Figure 1: Entity-relationship diagram.

2. Consider the entity-relationship diagram in Figure 1.

   (a) (5 points) Translate the entity-relationship diagram into SQL following the translation rules given in the lecture. Use "`TEXT`" as the domain of the columns. Make sure that the design does not allow null values. The code should run for PostgreSQL version 13 and above (try it).

   (b) (2 points) Describe in English an original real world example to which the entity-relationship diagram in Figure 1 could correspond. Give the real world meaning of entity sets E_1 and E_2, of relationship set S, and of attributes A, B, C, D, F, G, H, J, and K. Justify the candidate keys and participation constraints in the entity-relationship diagram.

– END OF PAPER –