

```

1  -- Kevin Hance
2  -- DBMS (CPSC 321)
3  -- HW7 (hw7.sql)
4
5  DROP TABLE IF EXISTS border;
6  DROP TABLE IF EXISTS city;
7  DROP TABLE IF EXISTS province;
8  DROP TABLE IF EXISTS country;
9
10 CREATE TABLE country(
11     country_code VARCHAR(10),
12     country_name VARCHAR(50),
13     gdp INT UNSIGNED,
14     inflation FLOAT,
15     PRIMARY KEY (country_code)
16 );
17
18 CREATE TABLE province(
19     province_name VARCHAR(50),
20     country_code VARCHAR(10),
21     area FLOAT UNSIGNED,
22     PRIMARY KEY (province_name, country_code),
23     FOREIGN KEY (country_code) REFERENCES country (country_code)
24 );
25
26 CREATE TABLE city(
27     city_name VARCHAR(50),
28     province_name VARCHAR(50),
29     country_code VARCHAR(10),
30     population INT UNSIGNED,
31     PRIMARY KEY (city_name, province_name, country_code),
32     FOREIGN KEY (province_name, country_code) REFERENCES province (province_name,
33     country_code)
34 );
35
36 CREATE TABLE border(
37     country_code_1 VARCHAR(10),
38     country_code_2 VARCHAR(10),
39     border_length FLOAT UNSIGNED,
40     PRIMARY KEY (country_code_1, country_code_2),
41     FOREIGN KEY (country_code_1) REFERENCES country (country_code),
42     FOREIGN KEY (country_code_2) REFERENCES country (country_code)
43 );
44
45 INSERT INTO country VALUES ('OS', 'Oswaldo', 78000, 6.7);
46 INSERT INTO country VALUES ('RL', 'Renlandia', 54000, 7.7);
47 INSERT INTO country VALUES ('GN', 'Geneva', 65000, 2.1);
48
49 INSERT INTO province VALUES ('Oslodo', 'OS', 8712);
50 INSERT INTO province VALUES ('St. Janice', 'OS', 76650);
51
52 INSERT INTO province VALUES ('Huport', 'RL', 91030);
53 INSERT INTO province VALUES ('Flaubury', 'RL', 5690);
54
55 INSERT INTO province VALUES ('Antalens', 'GN', 72003);
56 INSERT INTO province VALUES ('Huport', 'GN', 54041);
57
58 INSERT INTO city VALUES ('Tesa', 'Oslodo', 'OS', 964);
59 INSERT INTO city VALUES ('Sluurgan', 'Oslodo', 'OS', 5919);
60 INSERT INTO city VALUES ('Slaren', 'Oslodo', 'OS', 590190);
61 INSERT INTO city VALUES ('Belogonia', 'St. Janice', 'OS', 97635);
62 INSERT INTO city VALUES ('Britano', 'St. Janice', 'OS', 33434);
63
64 INSERT INTO city VALUES ('Piolas', 'Huport', 'RL', 46626);
65 INSERT INTO city VALUES ('Stombus', 'Huport', 'RL', 49384);
66 INSERT INTO city VALUES ('Outling', 'Flaubury', 'RL', 127);
67 INSERT INTO city VALUES ('Whita', 'Flaubury', 'RL', 52743);
68 INSERT INTO city VALUES ('Gonba', 'Flaubury', 'RL', 53146);

```

```

69 INSERT INTO city VALUES ('Blumore', 'Antalens', 'GN', 540000);
70 INSERT INTO city VALUES ('Juefbert', 'Antalens', 'GN', 120);
71 INSERT INTO city VALUES ('Tesa', 'Huport', 'GN', 4104300);
72 INSERT INTO city VALUES ('Stombus', 'Huport', 'GN', 1329900);
73
74 INSERT INTO border VALUES ('OS', 'GN', 4800);
75 INSERT INTO border VALUES ('GN', 'RL', 4500);
76 INSERT INTO border VALUES ('RL', 'OS', 1200);
77
78
79 -- reading question 3
80
81 DROP TABLE IF EXISTS border;
82 DROP TABLE IF EXISTS city;
83 DROP TABLE IF EXISTS province;
84 DROP TABLE IF EXISTS country;
85
86 CREATE TABLE country(
87     country_code VARCHAR(10),
88     country_name VARCHAR(50),
89     gdp INT UNSIGNED,
90     inflation FLOAT,
91     PRIMARY KEY (country_code)
92 );
93
94 CREATE TABLE province(
95     province_name VARCHAR(50),
96     country_code VARCHAR(10),
97     area FLOAT UNSIGNED,
98     PRIMARY KEY (province_name, country_code),
99     FOREIGN KEY (country_code) REFERENCES country (country_code)
100 );
101
102 CREATE TABLE city(
103     city_name VARCHAR(50),
104     province_name VARCHAR(50),
105     country_code VARCHAR(10),
106     population INT UNSIGNED,
107     PRIMARY KEY (city_name, province_name, country_code),
108     FOREIGN KEY (province_name, country_code) REFERENCES province (province_name,
109     country_code)
110 );
111
112 CREATE TABLE border(
113     country_code_1 VARCHAR(10),
114     country_code_2 VARCHAR(10),
115     border_length FLOAT UNSIGNED,
116     PRIMARY KEY (country_code_1, country_code_2),
117     FOREIGN KEY (country_code_1) REFERENCES country (country_code),
118     FOREIGN KEY (country_code_2) REFERENCES country (country_code)
119 );
120
121 INSERT INTO country VALUES ('OS', 'Oswaldo', 78000, 6.7);
122 INSERT INTO country VALUES ('RL', 'Renlandia', 54000, 7.7);
123 INSERT INTO country VALUES ('GN', 'Geneva', 65000, null);
124
125 SELECT count(c.inflation)
126 FROM country c
127 GROUP BY country_code
128 HAVING count(DISTINCT c.country_code) > 0.0;
129
130 -- TECHNICAL WORK
131 -- question 1
132 SELECT c.country_name, c.country_code, c.GDP, c.inflation, sum(cit.population)
133 FROM country c JOIN city cit ON cit.country_code = c.country_code
134 GROUP BY c.country_code;
135
136 -- question 2

```

```

137 set @population = 5000000;
138 SELECT p.country_code, p.province_name, p.area, sum(c.population)
139 FROM province p JOIN city c ON p.province_name = c.province_name AND p.country_code =
    c.country_code
140 GROUP BY p.province_name
141 HAVING sum(c.population) > @population;
142
143 -- question 3
144 SELECT c.country_code, c.country_name, count(DISTINCT cit.city_name)
145 FROM country c JOIN city cit ON c.country_code = cit.country_code
146 GROUP BY c.country_code
147 ORDER BY count(DISTINCT cit.city_name) DESC;
148
149 -- question 4
150 SELECT c.country_code, c.country_name, sum(p.area)
151 FROM country c JOIN province p ON c.country_code = p.country_code
152 GROUP BY c.country_code
153 ORDER BY sum(p.area) DESC;
154
155 -- question 5
156 set @min_cities = 5;
157 set @min_provinces = 1;
158 SELECT country_name
159 FROM country c JOIN province p JOIN city cit ON c.country_code = p.country_code AND
    p.country_code = cit.country_code
160 GROUP BY c.country_code
161 HAVING count(DISTINCT cit.city_name) >= @min_cities AND count(DISTINCT p.province_name)
    >= @min_provinces;
162
163 -- question 6
164 set @gdp = 60000;
165 SELECT c.country_code, c.gdp, sum(p.area)
166 FROM country c JOIN province p ON c.country_code = p.country_code
167 GROUP BY c.country_code
168 HAVING c.gdp >= @gdp
169 ORDER BY
170 CASE WHEN sum(p.area) <> 0 THEN sum(p.area) END DESC,
171 CASE WHEN c.gdp <> 0 THEN c.gdp END DESC;
172
173 -- question 7
174 DROP VIEW IF EXISTS sym_borders;
175 CREATE VIEW sym_borders AS
176 SELECT *
177 FROM border;
178 INSERT INTO sym_borders
179 SELECT country_code_2, country_code_1, border_length
180 FROM border;
181
182 SELECT * FROM sym_borders;
183
184 -- question 8
185 -- original
186 (SELECT c1.country_name, c2.country_name as c_gdp_high
187 FROM country c1 JOIN country c2 JOIN border bord
188 ON (bord.country_code_1 = c1.country_code AND bord.country_code_2 = c2.country_code)
189 WHERE c1.gdp < c2.gdp
190 AND c1.inflation > c2.inflation)
191 UNION
192 (SELECT c2.country_name, c1.country_name as c_gdp_high
193 FROM country c1 JOIN country c2 JOIN border bord
194 ON (bord.country_code_1 = c1.country_code AND bord.country_code_2 = c2.country_code)
195 WHERE c2.gdp < c1.gdp
196 AND c2.inflation > c1.inflation);
197 -- new query
198 SELECT c1.country_name, c2.country_name as c_gdp_high
199 FROM country c1 JOIN country c2 JOIN sym_borders bord
200 ON (bord.country_code_1 = c1.country_code AND bord.country_code_2 = c2.country_code)
201 WHERE (c1.gdp < c2.gdp
202 AND c1.inflation > c2.inflation)

```

```

203 OR (c2.gdp < c1.gdp
204 AND c2.inflation > c1.inflation);
205
206 -- question 9
207 SELECT c1.country_name, avg(c2.gdp), avg(c2.inflation)
208 FROM country c1 JOIN country c2 JOIN sym_borders b
209 ON b.country_code_1 = c1.country_code
210 AND b.country_code_2 = c2.country_code
211 GROUP BY b.country_code_1
212 ORDER BY
213 CASE WHEN avg(c2.gdp) <> 0 THEN avg(c2.gdp) END ASC,
214 CASE WHEN avg(c2.inflation) <> 0 THEN avg(c2.inflation) END ASC;
215
216 -- question 10
217 -- part 1: Show all cities which are in a country with a designated relationship between
218 -- the country's inflation and gdp, and those of a bordering country. Should
return
219 -- the name of the city, name of the province the city is in, and the
population of the
220 -- city, ordered by population of the city going from highest to lowest.
221 -- first argument: country 1 has a higher gdp, higher inflation
222 SELECT c.city_name, c.province_name, c.population
223 FROM country c1 JOIN country c2 JOIN sym_borders b JOIN city c
224 ON b.country_code_1 = c1.country_code
225 AND b.country_code_2 = c2.country_code
226 AND c.country_code = c1.country_code
227 WHERE c1.gdp > c2.gdp
228 AND c1.inflation > c2.inflation
229 ORDER BY c.population DESC;
230 -- second argument: country 1 has a higher gdp, lower inflation
231 SELECT c.city_name, c.province_name, c.population
232 FROM country c1 JOIN country c2 JOIN sym_borders b JOIN city c
233 ON b.country_code_1 = c1.country_code
234 AND b.country_code_2 = c2.country_code
235 AND c.country_code = c1.country_code
236 WHERE c1.gdp < c2.gdp
237 AND c1.inflation > c2.inflation
238 ORDER BY c.population DESC;
239
240 -- part 2: Show all provinces with a designated relationship between the area of that
province
241 -- and that of a province in a bordering country. Should return the name of
each province,
242 -- the area of each province, and the names of the countries that border each
other.
243 -- first argument: province 1 has a higher area
244 SELECT p1.province_name, p2.province_name, p1.area, p2.area, c1.country_code,
c2.country_code
245 FROM country c1 JOIN country c2 JOIN sym_borders b
246 JOIN province p1 JOIN province p2
247 ON b.country_code_1 = c1.country_code
248 AND b.country_code_2 = c2.country_code
249 AND c1.country_code = p1.country_code
250 AND c2.country_code = p2.country_code
251 WHERE p1.area > p2.area
252 ORDER BY p1.area DESC;
253 -- second argument: province 1 has a lower area
254 SELECT p1.province_name, p2.province_name, p1.area, p2.area, c1.country_code,
c2.country_code
255 FROM country c1 JOIN country c2 JOIN sym_borders b
256 JOIN province p1 JOIN province p2
257 ON b.country_code_1 = c1.country_code
258 AND b.country_code_2 = c2.country_code
259 AND c1.country_code = p1.country_code
260 AND c2.country_code = p2.country_code
261 WHERE p1.area < p2.area
262 ORDER BY p1.area DESC;
263

```