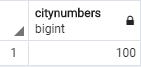
1. In wie vielen verschiedenen afrikanischen Städten gibt es eine Universität



|  |
| --- |
| SELECT count( DISTINCT ci.name ) AS cityNumbers FROM university u INNER JOIN city ci ON u.islocatedin = ci.id  INNER JOIN country cou ON ci.ispartof = cou.id  INNER JOIN continent con ON cou.ispartof = con.id  WHERE con.name = 'Africa' |

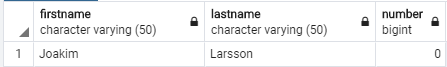
Anzahl der Ergebnistupel: 1



1. Wie viele Forenbeiträge (Posts) hat die älteste Person verfasst (Ausgabe: Name, #Forenbeiträge)?

|  |
| --- |
| SELECT p.firstname , p.lastname , COALESCE(COUNT ( post.id )) as number  FROM post Right JOIN person p ON post.creator= p.id WHERE p.birthday = ( SELECT MIN ( birthday ) FROM person ) GROUP BY firstname , lastname; |

Anzahl der Ergebnistupel: 1

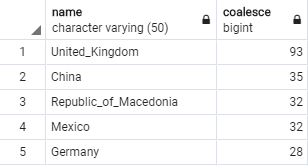


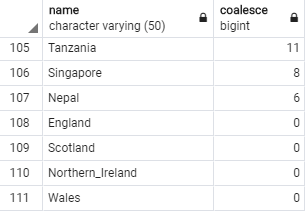
3. Wie viele Kommentare zu Posts gibt es aus jedem Land (Ausgabe aufsteigend sortiert nach Kommentaranzahl)?

Die Liste soll auch Länder enthalten, für die keine Post-Kommentare existieren, d.h. die Kommentaranzahl = 0 ist! (Funktion Coalesce)

|  |
| --- |
| SELECT cou.name, COALESCE ( COUNT(co.id), 0) FROM comment co RIGHT JOIN country cou  ON co.islocatedin = cou.id GROUP BY cou.name ORDER BY COALESCE DESC |

Anzahl der Ergebnistupel: 111

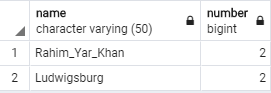




4. Aus welchen Städten stammen die meisten Nutzer (Ausgabe Name + Einwohnerzahl)?

|  |
| --- |
| SELECT d.name,COUNT(\*) AS number   FROM person p INNER JOIN city d ON p.islocatedin = d.id  GROUP BY d.name  HAVING COUNT(p.id) = (SELECT MAX(pcnt)  FROM ( SELECT COUNT(id) AS pcnt  FROM person   GROUP BY islocatedin ) AS cnt ) |

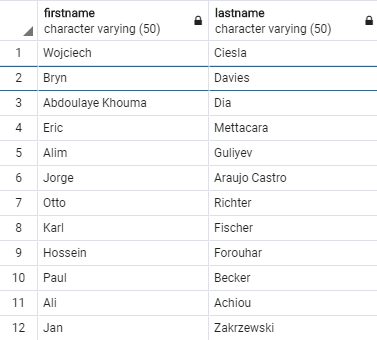
Anzahl der Ergebnistupel: 2



5. Mit wem ist ‘Hans Johansson’ befreundet?

|  |
| --- |
| WITH friendOfHans(hansFriend,hans) AS  (  SELECT userId2,userId1   FROM friendship\_View   WHERE userId1 = ( SELECT id  FROM person   WHERE firstname = 'Hans' AND lastname = 'Johansson' ))    SELECT firstname, lastname  FROM person  WHERE id IN (  SELECT hansFriend FROM friendOfHans ) |

Anzahl der Ergebnistupel: 12

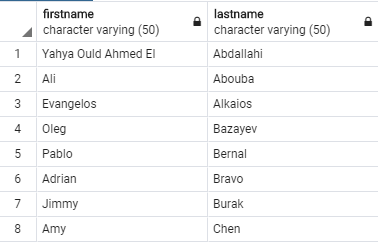


6. Wer sind die “echten” Freundesfreunde von ‘Hans Johansson’? “Echte” Freundesfreunde dürfen nicht gleichzeitig direkte Freunde von ‘Hans Johansson’ sein.

Sortieren Sie die Ausgabe alphabetisch nach dem Nachnamen.

|  |
| --- |
| WITH friendOfHans(hansFriend,hans) AS  (  SELECT userId2,userId1   FROM friendship\_View   WHERE userId1 = ( SELECT id  FROM person   WHERE firstname = 'Hans' AND lastname = 'Johansson' ))   SELECT firstname,lastname  FROM person  WHERE id  IN (SELECT f.userId2  FROM friendOfHans fOh, friendship\_View f WHERE fOh.hansFriend = f.userId1 AND NOT EXISTS (    SELECT fv.userId2  FROM friendship\_View fv  WHERE fv.userId1 = f.userId2 AND fv.userId2 IN (fOh.hans) )) AND NOT(firstname = 'Hans' AND lastname = 'Johansson') order by lastname |

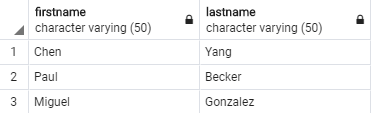
Anzahl der Ergebnistupel: 47



7. Welche Nutzer sind Mitglied in allen Foren, in denen auch ‘Mehmet Koksal’ Mitglied ist (Angabe Name)?

|  |
| --- |
| SELECT p.firstname, p.lastname   FROM person p   WHERE p.firstname <> 'Mehmet' AND p.lastname <> 'Mehmet' AND NOT EXISTS (  ( SELECT forumid   FROM hasmember  WHERE hasmember.persid = (  SELECT id   FROM person   WHERE firstname = 'Mehmet' AND lastname = 'Koksal' ) )  EXCEPT  ( SELECT forumid  FROM hasmember  WHERE hasmember.persid = p.id )  ) |

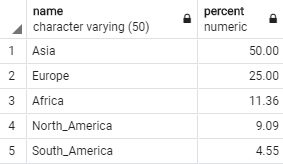
Anzahl der Ergebnistupel: 3



8. Geben Sie die prozentuale Verteilung der Nutzer bzgl. ihrer Herkunft aus verschiedenen Kontinenten an!

|  |
| --- |
| SELECT continent.name, ROUND(100\*cast(COUNT(person.id) as decimal)/(Select Count(person.id) From person),2) as percent from city JOIN person ON city.id = person.islocatedin JOIN country ON city.ispartof=country.id JOIN continent ON country.ispartof=continent.id GROUP BY continent.name ORDER by percent desc; |

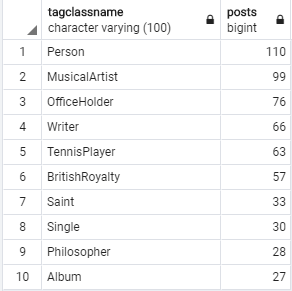
Anzahl der Ergebnistupel: 5



9. Zu welchen Themen (‘tag classes’) gibt es die meisten Posts? Geben Sie die Namen der Top 10 ‘tag classes’ mit ihrer Häufigkeit aus!

|  |
| --- |
| Select tagclass.tagclassname, count(post.id) as posts from tagclass JOIN hastype ON hastype.tagclassid = tagclass.id JOIN tag ON tag.id = hastype.tagid JOIN posthastag ON posthastag.tagid = tag.id JOIN post ON post.id = posthastag.postid GROUP BY tagclass.tagclassname ORDER BY posts desc limit 10; |

Anzahl der Ergebnistupel: 10



10. Welche Personen haben noch nie ein “Like” für einen Kommentar oder Post bekommen? Sortieren Sie die Ausgabe alphabetisch nach dem Nachnamen.

|  |
| --- |
| SELECT p.lastname, p.firstname  FROM person p WHERE p.id NOT IN   (Select po.creator   FROM person p  JOIN likespost l ON p.id = l.personid  JOIN post po ON l.postid = po.id) AND p.id NOT IN  (Select co.creator   FROM person p  JOIN likescomment l ON p.id = l.personid  JOIN comment co ON l.commentid = co.id) AND (p.id IN (  SELECT post.creator  FROM post) OR p.id IN(SELECT comment.creator FROM comment)  ) ORDER BY p.lastname; |

Anzahl der Ergebnistupel: 27



11. Welche Foren enthalten mehr Posts als die durchschnittliche Anzahl von Posts in Foren (Ausgabe alphabetisch sortiert nach Forumtitel)?

|  |
| --- |
| SELECT f.title, COUNT(p.id) from forum f JOIN post p ON p.forumid = f.id GROUP BY f.title HAVING COUNT(p.id) > (SELECT SUM(posts) from (SELECT COUNT(p.id) as posts from forum f  JOIN post p ON p.forumid = f.id) as aggregate)/(SELECT COUNT(f.id) FROM forum f) ORDER BY f.title; |

Anzahl der Ergebnistupel: 359



12. Welche Personen sind mit der Person befreundet, die die meisten Likes auf einen Post bekommen hat? Sortieren Sie die Ausgabe alphabetisch nach dem Nachnamen.

|  |
| --- |
| SELECT p.id, p.firstname, p.lastname FROM person p  JOIN friendship\_View k ON p.id = k.userId2 WHERE k.userId1 =  (Select p.id FROM person p  JOIN post m ON m.creator=p.id  JOIN likespost l ON m.id = l.postid  GROUP BY p.id  HAVING COUNT(l.personid) = (SELECT MAX(likedby) FROM   (Select p.id, COUNT(l.personid) as likedby FROM person p  JOIN post m ON m.creator=p.id  JOIN likespost l ON m.id = l.postid  GROUP BY p.id ORDER BY likedby desc) as x)) ORDER BY p.lastname asc; |

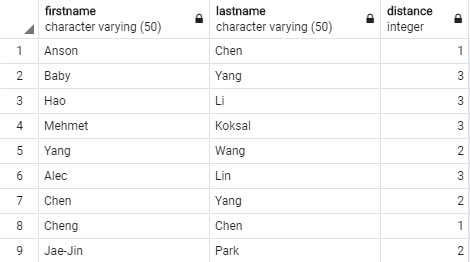
Anzahl der Ergebnistupel: 9



13. Welche Personen sind direkt oder indirekt mit ‘Jun Hu’ (id 94) verbunden (befreundet)? Geben Sie für jede Person die Distanz zu Jun an.

|  |
| --- |
| WITH RECURSIVE inrelation(id, firstname, lastname, distance, path) AS (  SELECT p1.id, p1.firstname, p1.lastname, 1, ARRAY[p1.id]  FROM friendship\_View fV JOIN person p1 ON fV.userId1 = p1.id  WHERE fV.userId2 = 94  UNION ALL  SELECT DISTINCT ON (p1.id) p1.id, p1.firstname, p1.lastname,   (inrelation.distance + 1), inrelation.path || p1.id  FROM inrelation, friendship\_View fV   JOIN person p1 ON fV.userId1 = p1.id  WHERE ((fV.userId2 = inrelation.id) AND NOT(p1.id = ANY(inrelation.path) ) AND NOT (p1.id=94) ) )  SELECT DISTINCT ON (id) firstname, lastname, distance FROM inrelation WHERE NOT id = 94 ORDER BY id, distance; |

Anzahl der Ergebnistupel: 72



14. Erweitern Sie die Anfrage zu Aufgabe 13 indem Sie zusätzlich zur Distanz den Pfad zwischen den Nutzern ausgeben.

|  |
| --- |
| WITH RECURSIVE inrelation(id, firstname, lastname, distance, path, nicepath) AS (  SELECT p1.id, p1.firstname, p1.lastname, 1, ARRAY[p1.id], ARRAY[p1.firstname || ' ' || p1.lastname || ' -> ' || 'Jun Hu']  FROM friendship\_View fV JOIN person p1 ON fV.userId1 = p1.id  WHERE fV.userId2 = 94  UNION ALL  SELECT DISTINCT ON (p1.id) p1.id, p1.firstname, p1.lastname,  (inrelation.distance + 1), inrelation.path || p1.id, (p1.firstname || ' ' || p1.lastname || ' -> '|| inrelation.nicepath)  FROM inrelation, friendship\_View fV  JOIN person p1 ON fV.userId1 = p1.id  WHERE ((fV.userId2 = inrelation.id) AND NOT(p1.id = ANY(inrelation.path) ) AND NOT (p1.id=94) AND NOT (p1.id IN (inrelation.id) ) )  )  SELECT DISTINCT ON (id) firstname, lastname, distance, array\_to\_string(nicepath, ' ') as path  FROM inrelation  ORDER BY id, distance ; |

Anzahl der Ergebnistupel: 72

