SQL - Social-Network Query Exercises

core-set.sql

```sql
-- 1. Find the names of all students who are friends with someone named Gabriel.

SELECT H1.name
FROM Highschooler H1
INNER JOIN Friend ON H1.ID = Friend.ID1
INNER JOIN Highschooler H2 ON H2.ID = Friend.ID2
WHERE H2.name = "Gabriel";


-- 2. For every student who likes someone 2 or more grades younger than themselves, return that

SELECT H1.name, H1.grade, H2.name, H2.grade
FROM Highschooler H1
INNER JOIN Likes ON H1.ID = Likes.ID1
INNER JOIN Highschooler H2 ON H2.ID = Likes.ID2
WHERE (H1.grade - H2.grade) >= 2;


-- 3. For every pair of students who both like each other, return the name and grade of both st

SELECT H1.name, H1.grade, H2.name, H2.grade
FROM Highschooler H1, Highschooler H2, Likes L1, Likes L2
WHERE (H1.ID = L1.ID1 AND H2.ID = L1.ID2) AND (H2.ID = L2.ID1 AND H1.ID = L2.ID2) AND H1.name <
ORDER BY H1.name, H2.name;


-- 4. Find all students who do not appear in the Likes table (as a student who likes or is like

SELECT name, grade
FROM Highschooler
WHERE ID NOT IN (
  SELECT DISTINCT ID1
  FROM Likes
  UNION
  SELECT DISTINCT ID2
  FROM Likes

)
ORDER BY grade, name;
```

```sql

-- 5. For every situation where student A likes student B, but we have no information about whe

SELECT H1.name, H1.grade, H2.name, H2.grade
FROM Highschooler H1
INNER JOIN Likes ON H1.ID = Likes.ID1
INNER JOIN Highschooler H2 ON H2.ID = Likes.ID2
WHERE (H1.ID = Likes.ID1 AND H2.ID = Likes.ID2) AND H2.ID NOT IN (
  SELECT DISTINCT ID1
  FROM Likes
);


-- 6. Find names and grades of students who only have friends in the same grade. Return the res

SELECT name, grade
FROM Highschooler H1
WHERE ID NOT IN (
  SELECT ID1
  FROM Friend, Highschooler H2
  WHERE H1.ID = Friend.ID1 AND H2.ID = Friend.ID2 AND H1.grade <> H2.grade
)
ORDER BY grade, name;


-- 7. For each student A who likes a student B where the two are not friends, find if they have

SELECT DISTINCT H1.name, H1.grade, H2.name, H2.grade, H3.name, H3.grade
FROM Highschooler H1, Highschooler H2, Highschooler H3, Likes L, Friend F1, Friend F2
WHERE (H1.ID = L.ID1 AND H2.ID = L.ID2) AND H2.ID NOT IN (
  SELECT ID2
  FROM Friend
  WHERE ID1 = H1.ID
) AND (H1.ID = F1.ID1 AND H3.ID = F1.ID2) AND (H2.ID = F2.ID1 AND H3.ID = F2.ID2);


-- 8. Find the difference between the number of students in the school and the number of differ

SELECT COUNT(*) - COUNT(DISTINCT name)
FROM Highschooler;


-- 9. Find the name and grade of all students who are liked by more than one other student.

SELECT name, grade
FROM Highschooler
INNER JOIN Likes ON Highschooler.ID = Likes.ID2
GROUP BY ID2
HAVING COUNT(*) > 1;
```

extras.sql

```sql
-- 1. For every situation where student A likes student B, but student B likes a different stu

SELECT H1.name, H1.grade, H2.name, H2.grade, H3.name, H3.grade
FROM Highschooler H1, Highschooler H2, Highschooler H3, Likes L1, Likes L2
WHERE H1.ID = L1.ID1 AND H2.ID = L1.ID2 AND (H2.ID = L2.ID1 AND H3.ID = L2.ID2 AND H3.ID <> H1


-- 2. Find those students for whom all of their friends are in different grades from themselve

SELECT name, grade
FROM Highschooler H1
WHERE grade NOT IN (
  SELECT H2.grade
  FROM Friend, Highschooler H2
  WHERE H1.ID = Friend.ID1 AND H2.ID = Friend.ID2
);


-- 3. What is the average number of friends per student? (Your result should be just one numbe

SELECT AVG(count)
FROM (
  SELECT COUNT(*) AS count
  FROM Friend
  GROUP BY ID1
);


-- 4. Find the number of students who are either friends with Cassandra or are friends of frie

SELECT COUNT(*)
FROM Friend
WHERE ID1 IN (
  SELECT ID2
  FROM Friend
  WHERE ID1 IN (
    SELECT ID
    FROM Highschooler
    WHERE name = 'Cassandra'
  )
);


-- 5. Find the name and grade of the student(s) with the greatest number of friends.

SELECT name, grade
FROM Highschooler
INNER JOIN Friend ON Highschooler.ID = Friend.ID1
GROUP BY ID1
HAVING COUNT(*) = (
```

```
  SELECT MAX(count)
  FROM (
    SELECT COUNT(*) AS count
    FROM Friend
    GROUP BY ID1
  )
);
```

modification.sql

```
-- 1. It's time for the seniors to graduate. Remove all 12th graders from Highschooler.

DELETE FROM Highschooler
WHERE grade = 12;



-- 2. If two students A and B are friends, and A likes B but not vice-versa, remove the Likes t

DELETE FROM Likes
WHERE ID2 IN (
  SELECT ID2
  FROM Friend
  WHERE Friend.ID1 = Likes.ID1
) AND ID2 NOT IN (
  SELECT L.ID1
  FROM Likes L
  WHERE L.ID2 = Likes.ID1
);

DELETE FROM Likes
WHERE ID1 IN (
  SELECT Likes.ID1
  FROM Friend
  INNER JOIN Likes USING(ID1)
  WHERE Friend.ID2 = Likes.ID2
) AND ID2 NOT IN (
  SELECT Likes.ID1
  FROM Friend
  INNER JOIN Likes USING(ID1)
  WHERE Friend.ID2 = Likes.ID2
);



-- 3. For all cases where A is friends with B, and B is friends with C, add a new friendship fo

INSERT INTO Friend
SELECT DISTINCT F1.ID1, F2.ID2
FROM Friend F1, Friend F2
WHERE F1.ID2 = F2.ID1 AND F1.ID1 <> F2.ID2 AND F1.ID1 NOT IN (

  SELECT F3.ID1
  FROM Friend F3
```

```
  WHERE F3.ID2 = F2.ID2
);

INSERT INTO Friend
SELECT F1.ID1, F2.ID2
FROM Friend F1
INNER JOIN Friend F2 ON F1.ID2 = F2.ID1
WHERE F1.ID1 <> F2.ID2
EXCEPT
SELECT * FROM Friend;
```

gaurang444 commented on Sep 18, 2017

Thanks for saving my lab exam

Tommytrungto commented on Mar 19, 2018

For the question #4, why do ID1 and ID2 have to be distinct?

coroche commented on Mar 29, 2019

Thanks for the great resource. Your solutions are a lot more elegant than what I've been coming up with. I'm not so sure on Extras Q4. I might be missing some subtleties but it seems like your solution doesn't count friends of Cassandra (only friends of friends) and counts Cassandra herself twice.

ibadlisham commented on Aug 27, 2019 • edited

This is great! It was very helpful but I would like to note that the some of the group by statements don't work when sql_mode = 'ONLY_FULL_GROUP_BY' and that I agree with coroche, the answer for #4 is not really answering the question even though the output is the same.

This was the answer that I came up with. I had to replace the **@cassandra** variable in the main query in order to work with the SQLite program.

SET **@cassandra**=(select id from highschooler where name='Cassandra');
select count(*) from (
select f1.id1
from friend f1
where f1.id2 = **@cassandra**
and f1.id1 != **@cassandra**
union
select f2.id2
from friend f2
where f2.id1 in (select f1.id1 from friend f1 where f1.id2 = **@cassandra**)
and f2.id2 != **@cassandra**
) a

rchatti commented on Feb 21, 2020

@backpackerhh Savior!
@ibadlisham... I tried this in Oracle for Extra # 4:

```
select count(distinct f1.ID2) + count(distinct f2.ID2)
from friend f1
inner join friend f2
on f1.ID2 = f2.ID1

inner join highschooler hs
on hs.id = f1.id1

inner join highschooler hs2
on f2.id2 = hs2.id

where f1.ID1 <> f2.ID2
and hs.name = 'Cassandra';
```

kirinzero13 commented on Feb 21, 2020 • edited

We must receive knowledge from unique sources. These may be books or educational resources. People often confuse the Internet and educational content. Content is the soul of the Internet but not its essence. I recently read an article about the possibilities of a child's cognitive education. You can use the game form to memorize words and sentences.

Cherisea commented on Feb 28, 2020

Here is my solution for Q4:

```
SELECT COUNT(ID2) FROM Friend WHERE ID1 IN (SELECT ID2 FROM Friend WHERE ID1 = (SELECT ID FROM Highschooler WHERE name = 'Cassandra') AND ID2 <> ID1);
```

But I am not sure how come the solution in Extras Q2 works. It seems that the SELECT statement in WHERE clause pulls out the grades of those who HAVE friends. How is this going to find out students whose friends are all from different grades?

Does anyone have the same misgiving? Thanks for any info or explanation!

Dhruv-Garg79 commented on Mar 13, 2020

(H1.ID = Likes.ID1 AND H2.ID = Likes.ID2) is redundant in 5th ques, because join already took care of this case.

menapertas commented on Jun 25, 2021

> Here is my solution for Q4:
> ```
> SELECT COUNT(ID2) FROM Friend WHERE ID1 IN (SELECT ID2 FROM Friend WHERE ID1 = (SELECT ID
> FROM Highschooler WHERE name = 'Cassandra') AND ID2 <> ID1);
> ```
>
> But I am not sure how come the solution in Extras Q2 works. It seems that the SELECT statement in WHERE clause pulls out the grades of those who HAVE friends. How is this going to find out students whose friends are all from different grades?
>
> Does anyone have the same misgiving? Thanks for any info or explanation!

your solution for Q4 is no difference than all other. even is sql is not answering, you could assume it does. if you dont change schema and keep inserting info, the result wont change. because cassanda is always is going to be friend of cassandra's fried. That's way answer is kind of short cut coorect. but if question was asking name of friends instead of numbe of friends, then it would change all thing. stil your solution not right thou

torchlooksgood commented on Feb 2 • edited

Here is my solution for Q9:

```
SELECT name, grade
FROM Highschooler
INNER JOIN Likes ON Highschooler.ID = Likes.ID2
GROUP BY ID2
HAVING COUNT("star symbol") IN (SELECT MAX(likee)
FROM (SELECT ID2, count(*) AS likee
FROM Likes
GROUP BY ID2));
```

santusam commented on Mar 15

how to find table list with last when it was used by someone and in last 5-6 months how many times that table has been used in queries in redshift