Final exam task 1 — Fundamentals of Databases

Dimitri Tabatadze · 17:54, January 10, 2024

1. (a) is implemented, since there's no *provides* relation and there is *orgID* attribute in reasearcher relation.
2. - The difference between (a) and (b) is that in (a) one reasearcher can only be provided by one organization, but one organization can provide many reasearchers.
   - The difference between (b) and (c) is that

     (a) in (b) we store an additional table *provide*

     (b) in (b) one reasearcher be provided by many organizations.
   - The difference between (a) and (c) is that in (a) we store an additional table *provide*.
3. (c) misses the aditional table *provide*.
4. There's no need for *waterType* since you can already tell the type of water by the *waterID*. The most significant digit of the *waterID* is 2 if the water type is standing and 1 if it's running.
5.
```
with funds as (
    select projectID, sum(amount) as availableFunds from finances
    group by projectID
)
select
    projectID,
    projectTitle,
    totalCost,
    ifnull(availableFunds, 0) as availableFunds,
    totalCost - ifnull(availableFunds, 0) as missingFunds
from project left join funds using (projectID)
having missingFunds > 0;
```

   Result:

projectID	projectTitle	totalCost	availableFunds	missingFunds
1	Research on sodium content of water bodies in Georgia.	100000	10000	90000
2	Improving water quality	2000000	115000	1885000
4	Endemic water species	20000	0	20000

6.
```
-- create or replace view avgWater as
with everyWater as (
    select ifnull(quality, 0) as quality from runningwater
    union all
    select ifnull(quality, 0) as quality from standingwater
)
select waterType, count(*) as amount, (select avg(quality) as totalAverage from
    everyWater) as totalAverage from (
    select "runningWater" as waterType, ifnull(quality, 0) as quality from runningwater
    union all
    select "standingWater" as waterType, ifnull(quality, 0) as quality from
        standingwater
) c
where quality > (select avg(quality) as totalAverage from everyWater)
GROUP BY waterType;
```

   Result:

waterType	amount	totalAverage
runningWater	6	5.8421
standingWater	5	5.8421