# SQL

## 1. Определить рейтинг отрезков по сумме расстояний до них от точек на рисунке? (теперь работает, но не факт что правильно)

USE sketching;

CREATE OR REPLACE VIEW segments AS (

SELECT object.idobject AS segment\_id,

point.x AS x,

point.y AS y

FROM object

INNER JOIN objpoints ON object.idobject = objpoints.idobject

INNER JOIN point ON objpoints.idpoint = point.idpoint

WHERE objtype = 1

);

DROP FUNCTION IF EXISTS calc\_distance;

DELIMITER //

CREATE FUNCTION calc\_distance (segment\_id INT, point\_id INT)

RETURNS FLOAT

READS SQL DATA

BEGIN

SELECT x, y INTO @x0, @y0

FROM `point`

WHERE point\_id = `point`.idpoint

LIMIT 0, 1;

SELECT x, y INTO @x1, @y1

FROM segments

WHERE segment\_id = segments.segment\_id

LIMIT 0, 1;

SELECT x, y INTO @x2, @y2

FROM segments

WHERE segment\_id = segments.segment\_id

LIMIT 1, 1;

SET @a = @y1 - @y2,

@b = @x2 - @x1,

@c = @x1 \* @y2 - @x2 \* @y1;

RETURN ABS(@a \* @x0 + @b \* @y0 + @c) / SQRT(@a \* @a + @b \* @b);

END //

DELIMITER ;

SELECT segments.segment\_id AS segment\_id,

SUM(calc\_distance(segments.segment\_id, point.idpoint)) AS rating

FROM segments, point

GROUP BY segments.segment\_id ORDER BY rating DESC;

## 2. Для каждой пары перпендикулярных отрезков определить тот из них, чья длина наибольшая

USE sketching;

SELECT hor\_id, hor\_len, vert\_id, vert\_len, hor\_layer AS layer,

CASE

WHEN hor\_len >= vert\_len THEN hor\_id

WHEN vert\_len > hor\_len THEN vert\_id

END AS longest

FROM (

(

SELECT DISTINCT entity.identity AS hor\_id, SUM(point.x)\*SUM(point.x) + SUM(point.y)\*SUM(point.y) AS hor\_len, constrtype.name AS `hor\_type`, entity.layer AS `hor\_layer`

FROM entity

JOIN constrinfo ON constrinfo.identity = entity.identity

JOIN `constraint` ON `constraint`.idconstraint = constrinfo.idconstraint

JOIN constrtype ON constrtype.idconstrtype = `constraint`.constrtype

JOIN object ON object.idobject = entity.identity

JOIN objtype ON objtype.idobjtype = object.objtype

JOIN objpoints ON objpoints.idobject = object.idobject

JOIN point ON point.idpoint = objpoints.idpoint

WHERE objtype.name = "Segment" AND constrtype.name = "Horizontal"

GROUP BY entity.identity

) AS hor

JOIN (

SELECT DISTINCT entity.identity AS vert\_id, SUM(point.x)\*SUM(point.x) + SUM(point.y)\*SUM(point.y) AS vert\_len, constrtype.name AS `vert\_type`, entity.layer AS `vert\_layer`

FROM entity

JOIN constrinfo ON constrinfo.identity = entity.identity

JOIN `constraint` ON `constraint`.idconstraint = constrinfo.idconstraint

JOIN constrtype ON constrtype.idconstrtype = `constraint`.constrtype

JOIN object ON object.idobject = entity.identity

JOIN objtype ON objtype.idobjtype = object.objtype

JOIN objpoints ON objpoints.idobject = object.idobject

JOIN point ON point.idpoint = objpoints.idpoint

WHERE objtype.name = "Segment" AND constrtype.name = "Vertical"

GROUP BY entity.identity

) AS vert

ON hor.hor\_layer = vert.vert\_layer

)

Альтернативное решение, которое завязывается на ограничение ортогональности и ебало рот приколы с горизонтальным и вертикальным. Почему? потому что если вы автоматом не накидываете ограничение на ортогональность для ограниченных вертикально и горизонтально отрезков - вы ебланы и вам запрещено пользоваться базами данных.

Сначала собираем в проекцию сегмент (его ид) и длину отрезка сегмента + выбираем среди них только те, что имею ограничение типа орт (ид ограничения тоже сейвим) - итого тройка объект-длина-ограничение

потом собираем объект (сегмент1, сегмент2, длина1, длина2, наидлиннейший) через джойн с самим собой.

А еще здесь не само расстояние, а его квадрат (просто не знал что есть функция квадратного корня в сиквеле и нахуй она нужна все еще не ебу)

## CREATE VIEW seglen AS

## SELECT DISTINCT a.idobject, (a.x - b.x) \* (a.x - b.x) + (a.y - b.y) \* (a.y - b.y) AS length, i.idconstraint FROM

## (SELECT \* FROM object o

## JOIN objpoints op ON o.idobject=op.idobject

## JOIN point p ON p.idpoint=op.idpoint

## WHERE o.objtype=1) AS a

## JOIN

## (SELECT \* FROM object o

## JOIN objpoints op ON o.idobject=op.idobject

## JOIN point p ON p.idpoint=op.idpoint

## WHERE o.objtype=1) AS b

## ON a.idpoint<>b.idpoint AND a.idobject=b.idobject

## JOIN constrinfo i ON i.identity=a.idobject

## JOIN `constraint` c ON c.idconstraint=i.idconstraint

## WHERE c.constrtype=7;

## 

## SELECT x.id1 as segment1, x.id2 AS segment2, x.s1l AS segment1len, x.s2l AS segment2len,

## CASE WHEN x.s1l>x.s2l THEN x.id1

## ELSE x.id2 END AS longest

## FROM (

## SELECT s1.idobject AS id1, s2.idobject AS id2, s1.length AS s1l, s2.length AS s2l

## FROM seglen s1

## JOIN seglen s2 ON s1.idconstraint=s2.idconstraint AND s1.idobject<>s2.idobject

## ) as x;

## 

## DROP VIEW seglen;

## 3. Определить рейтинг типов графических объектов по их количеству на каждом слое.

а

## 4. Определить количество окружностей, связанных ограничениями с равными отрезками (скорее всего неправильно)

Если что тут ответ 0 просто потому что у нас нет объектов окружностей) - Коровкин рофлер. Решение вроде верное.

use sketching;

select count(id2) from

(

select e.identity as id, c.idconstraint as idcon from

entity e join constrinfo c on e.identity = c.identity

join object o on o.idobject = e.identity

join `constraint c1 on c1.idconstraint = c.idconstraint

where constrtype = 1 and objtype = 1

) as t1

join

(

select e2.identity as id2, idconstraint as idcon2 from

entity e2 join constrinfo c2 on e2.identity = c2.identity

join object o2 on o2.idobject = e2.identity

where objtype = 5

) as t2

on t1.idcon = t2.idcon2

## **5. Определить дуги, имеющие общие точки с отрезками** use sketching;

select distinct(idobject), name

from (

( select x, y from objtype

join object on objtype.idobjtype = object.objtype

join objpoints on objpoints.idobject = object.idobject

join point on point.idpoint = objpoints.idpoint

where (name = 'Segment') ) as table1

join (

select x, y, object.idobject, [objtype.name](https://vk.com/away.php?to=http%3A%2F%2Fobjtype.name&cc_key=) from objtype

join object on objtype.idobjtype = object.objtype

join objpoints on objpoints.idobject = object.idobject

join point on point.idpoint = objpoints.idpoint

where (name = 'Arc') ) as table2

on table1.x = table2.x and table1.y = table2.y

);

Вариант 2

*USE sketching;*

*SELECT arc\_table.arc\_id, arc\_table.arc\_layer, seg\_table.seg\_id, seg\_table.seg\_layer*

*FROM (*

*(*

*SELECT o.idobject AS arc\_id, e.layer AS arc\_layer, points.arc\_x, points.arc\_y*

*FROM object o*

*JOIN entity e*

*ON o.idobject = e.identity*

*JOIN (*

*SELECT p.x AS arc\_x, p.y AS arc\_y, op.idobject AS arc\_id\_point*

*FROM point p*

*JOIN objpoints op*

*ON p.idpoint = op.idpoint*

*) AS points*

*ON points.arc\_id\_point = o.idobject*

*JOIN objtype ot*

*ON o.objtype = ot.idobjtype*

*WHERE ot.name = "Arc"*

*) AS arc\_table*

*JOIN (*

*SELECT o.idobject AS seg\_id, e.layer AS seg\_layer, points.seg\_x, points.seg\_y*

*FROM object o*

*JOIN entity e*

*ON o.idobject = e.identity*

*JOIN (*

*SELECT p.x AS seg\_x, p.y AS seg\_y, op.idobject AS seg\_id\_point*

*FROM point p*

*JOIN objpoints op*

*ON p.idpoint = op.idpoint*

*) AS points*

*ON points.seg\_id\_point = o.idobject*

*JOIN objtype ot*

*ON o.objtype = ot.idobjtype*

*WHERE ot.name = "Segment"*

*) AS seg\_table*

*ON arc\_x = seg\_x AND arc\_y = seg\_y AND arc\_layer = seg\_layer*

*)*

## 6. Определить существуют ли объекты, такие что для нескольких из них в рамках одного слоя существуют одинаковые ограничения размера

select count(distinct (idobject)),constrtype.name

from `entity`, `layer`, `object`, `constrinfo`, `constrtype`, `constraint`

where entity.layer = layer.idlayer and entity.identity = object.idobject

and constrinfo.identity = entity.identity and constrinfo.idconstraint = `constraint`.idconstraint

and `constraint`.constrtype = constrtype.idconstrtype

andconstrtype.name = "Equal"

**Еще один вариант решения который учитывает нахождение на одном слое**

## ***-- Определить существуют ли объекты, такие что***

## ***-- для нескольких из них в рамках одного слоя***

## ***-- существуют одинаковые ограничения размера***

## 

## ***-- выбираем тройки объект-слой-ограничение такие что***

## ***-- ограничение имеет тип 1 (т.е. размера)***

## **CREATE VIEW olc AS**

## **SELECT o.idobject, e.layer, c.idconstraint FROM object o**

## **JOIN entity e ON e.identity=o.idobject**

## **JOIN constrinfo info ON info.identity=o.idobject**

## **JOIN `constraint` c ON c.idconstraint=info.idconstraint**

## **WHERE c.constrtype=1;**

## 

## ***-- попарно объединяем тройки если***

## ***-- \* не одна и так же запись***

## ***-- \* на одном слою***

## ***-- \* одно и то же ограничение размера***

## **SELECT EXISTS (**

## **SELECT \* FROM olc olc1**

## **JOIN olc olc2 ON**

## **olc1.layer=olc2.layer**

## **AND olc1.idconstraint=olc2.idconstraint**

## **AND olc1.idobject <> olc2.idobject);**

## 

## ***-- удаляем проекцию (или как эта хуйня называется)***

## **DROP VIEW olc;**

## 

## **7. Определить, есть ли на рисунке отрезок, перпендикулярный семи другим отрезкам?( где в решении проверяется 7?)**

Добавил проверку на 7, если то что внутри правильно, то все должно быть окей

SELECT COUNT(\*)

FROM (

SELECT \*

FROM (

(

SELECT entity.identity AS "segment id"

FROM entity

JOIN object

ON object.idobject = entity.identity

JOIN objtype

ON objtype.idobjtype = object.objtype

JOIN constrinfo

ON constrinfo.identity = entity.identity

JOIN `constraint`

ON `constraint`.idconstraint = constrinfo.idconstraint

JOIN constrtype

ON constrtype.idconstrtype = `constraint`.constrtype

WHERE constrtype.name = "Horizontal" AND objtype.name = "Segment"

) AS t1

JOIN (

SELECT count(entity.identity) AS orth\_num

FROM entity

JOIN object

ON object.idobject = entity.identity

JOIN objtype

ON objtype.idobjtype = object.objtype

JOIN constrinfo

ON constrinfo.identity = entity.identity

JOIN `constraint`

ON `constraint`.idconstraint = constrinfo.idconstraint

JOIN constrtype

ON constrtype.idconstrtype = `constraint`.constrtype

WHERE constrtype.name = "Vertical" AND objtype.name = "Segment"

) AS t2

)

)

WHERE orth\_num >= 7

## **8. Определить все объекты, связанные вертикальным ограничением**

USE sketching;

SELECT object.idobject AS id, objtype.name AS "object type"

FROM object

JOIN objtype

ON objtype.idobjtype = object.objtype

JOIN entity

ON entity.identity = object.idobject

JOIN constrinfo

ON constrinfo.identity = entity.identity

JOIN `constraint`

ON `constraint`.idconstraint = constrinfo.idconstraint

JOIN constrtype

ON constrtype.idconstrtype = `constraint`.constrtype

WHERE constrtype.name = "Vertical"

2 вариант

select COUNT(DISTINCT idobject) from object

join entity e on object.idobject = e.identity

join constrinfo c on e.identity = c.identity

join `constraint` c2 on c2.idconstraint = c.idconstraint

where constrtype = 2;

**9. Среди отрезков, касающихся дуг окружностей найти тот, чья длина наибольшая.**

# **XQuery**

## **1. По скольким различным странам протекают реки, впадающие в океан?**

let $d := doc("factbook.xml")/mondial

let $riverlocated :=distinct-values(

for $i in $d/river

where $i/to/@type = "sea"

return data($i/located/@country)

)

return count($riverlocated)

Длинный

let $d := doc("factbook.xml")/mondial

let $riverlocated :=distinct-values(

for $i in $d/river

where $i/to/@type = "sea"

return data($i/located/@country)

)

let $countrys := (

for $i in $d/country, $j in $riverlocated

where $i/@id = $j

return data($i/name)

)

let $ans := count($countrys)

return $ans

## **2. Определить страны, участвующие не менее чем в трех международных организациях и граничащие друг с другом?**

let $d := doc("factbook.xml")/mondial  
let $uniqMembers := distinct-values( $d/organization/members/@country)  
  
let $countryGood := (for $country in $uniqMembers  
let $cnt := count($d//members[@country = $country])  
where $cnt >= 3  
return $country )  
  
let $res := distinct-values(for $i in $countryGood, $j in $countryGood  
where $d/country[@id = $i]/border[@country = $j]  
return $d/country[@id = $i]/@name)  
return $res

Можно переменную res заменить прям на пары стран (остальное такое же), но тогда уникальность хз как сделать (чтобы из пар <а, б> <б, а> была только одна)

let $d := doc("factbook.xml")/mondial

let $uniqMembers := distinct-values( $d/organization/members/@country)

let $countryGood := (

for $country in $uniqMembers

where count($d//members[@country = $country]) >= 3

return $country

)

let $res := (

for $i in $countryGood, $j in $countryGood

where $d/country[@id = $i]/border[@country = $j]

return

<pair\_c>

<first>

{$d/country[@id = $i]/@name}

</first>

<second>

{$d/country[@id = $j]/@name}

</second>

</pair\_c>

)

return $res

## **3. Определить страны с наибольшей суммарной длиной рек, протекающих по данным странам.**

let $doc := doc('factbook.xml')

let $countries := $doc//country

let $locations := $doc//river/located

let $stats:= (

for $country in $countries

return

<Country>

{$country/name/text()}

<lengths>

{

sum(

$locations[@country = $country/@id]/../@length/data()

)

}

</lengths>

</Country>

)

return $stats[lengths = max($stats/lengths)]/text()

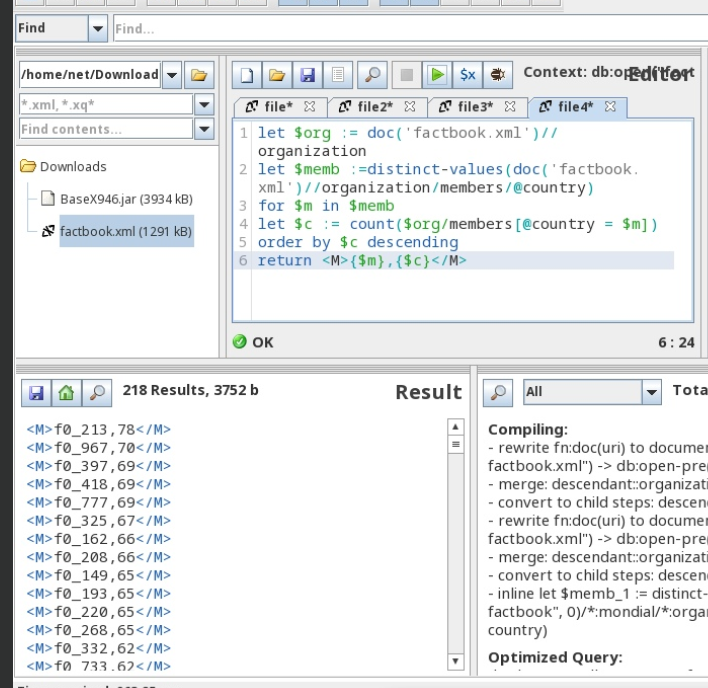
## **4. Определить для страны с минимальной площадью количество населения, проживающего в городах**

let $d := doc("factbook.xml")/mondial

let $min\_country := $d/country[@total\_area=min($d/country/@total\_area)]

return sum($min\_country//city/population)

## **5. Определить страны - участники наибольшего числа международных организаций.**



Вариант на скрине не работает. Мой вариант:

let $d := doc("factbook.xml")

let $org := $d//organization

let $memb\_ids := distinct-values($org/members/@country)

let $countries:=(

for $m in $memb\_ids

return

<body>

{data($d/mondial/country[@id=$m]/@name)}

<numb\_org>

{count($org/members[@country=$m])}

</numb\_org>

</body>

)

return $countries[numb\_org = max($countries/numb\_org)]

## 

## 6. Определить суммарное количество населения, проживающего в городах, расположенных в провинциях с наименьшим количеством населения, в каждой стране

let $minpro:= (

for $coun in doc("factbook.xml")//country

for $i in $coun/province

where $i/@population/data() = min($coun/province/@population/data())

return <p country="{$coun/name/data()}"> {sum($i/city/population/data())} </p>

)

return $minpro

## 7. Определить страны, граничащие с максимальным количеством других стран

let $countries := doc('factbook.xml')//country

let $stats := (

for $country in $countries

return <country>

{$country/name/text()}

<borders>

{count($country/border)}

</borders>

</country>

)

return $stats[borders = max($stats/borders)]/text()

(если надо список по убыванию вывести, то вместо последней строчки вот это:

for $x in $stats

order by $x/borders cast as xs:int descending

return $x

## 8. Определить страны, в которых население какого-либо города превышает население какой-либо провинции

for $country in doc("factbook.xml")//country

let $cityMax:=max($country/city/population)

let $provinceMin:=min($country/province/@population)

return if($cityMax>$provinceMin) then data($country/name) else ()

## **9. Определить города, расположенные на реках, впадающих в другие реки**

let $d := doc("factbook.xml")/mondial

let $river\_to\_river := distinct-values(

for $i in $d/river

where $i/to/@type="river"

return $i/@id

)

for $c in $d/country/city, $r in $d/river

where $c/located\_at[@type="river"] and

data($c/located\_at[@type="river"]/@water) = $r/@id and

$r/to[@type="river"]

return data($c/name)

## **10. Определить количество людей, проживающих в трех городах с наибольшим количеством жителей**

let $d := doc("factbook.xml")/mondial

let $cities := $d//city

let $cities\_where\_population\_not\_null := (for $i in $cities

where $i[population]

return $i)

let $ordered\_cities := (for $i in $cities\_where\_population\_not\_null

order by $i/population[last()] cast as xs:long descending

return $i)

return $ordered\_cities[position()<4]/population[last()]/data()

## 

## **11.Определить страны, имеющие наибольшую разницу в населении входящих в них провинций**

let $d := doc("factbook.xml")/mondial

let $countries := $d/country

let $countries\_with\_provinces :=

(for $i in $countries

where $i/province[@population]

return $i)

let $res :=

(for $i in $countries\_with\_provinces

return

<country>

<name>

{$i/name}

</name>

<maxDifference>

{(max($i/province/@population/data()) - min($i/province/@population/data())) cast as xs:long}

</maxDifference>

</country>)

let $ordered\_res := (for $i in $res

order by $i//maxDifference cast as xs:long descending

return $i)

return $ordered\_res[position()<2]//name/text()

**2 вариант(работает, пока не гарантируется правильность)**

**(Если задание все таки в разнице суммарной популяции в провинциях, то этот)**

let $stats := (

for $country in doc("factbook.xml")//country

order by sum($country/province/@population) descending

return <country>

{$country/name/text()}

<sum>

{sum($country/province/@population)}

</sum>

</country>

)[1]

let $maxCountry := $stats/text()

for $country in doc("factbook.xml")//country

let $provinceSum:=sum($country/province/@population)

return if ($provinceSum=0) then

(<countries>

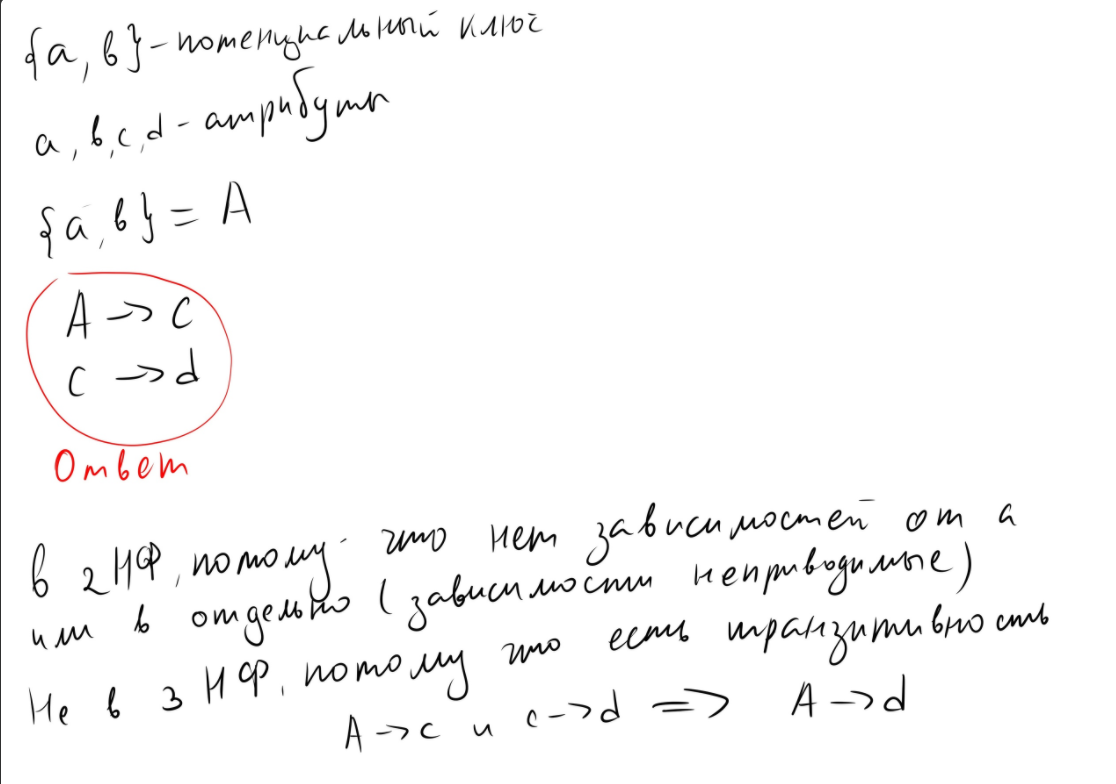
<country1>{$country/name/text()}</country1>

<country2>{$maxCountry}</country2>

</countries>)

# **ДОП ВОПРОСЫ**

## 1. Привести пример отношения, находящегося в 2НФ, но не находящегося в 3НФ.

В отношении 4 атрибута: a, b, c, d. a и b входят в состав потенциального ключа. Определить набор функциональных зависимостей, которые обеспечат нахождение отношения в 2нф, но не в 3нф ****

## 2. Создать таблички с о связями на MySQL. Он диаграмму скинет и скажет конкретное задание. У меня было - создать таблицу “кафедра” и “сотрудники” так, что у любой кафедры всегда должен быть один заведующий(из числа сотрудников этой кафедры), но на некоторых кафедрах может не быть сотрудников. (то есть между сотрудниками и кафедрами две связи: одна - заведует, вторая - состоит).

**1 Вариант**

CREATE TABLE cafedra (

caf\_id INT,

name VARCHAR(45) NOT NULL UNIQUE,

manager\_id INT NOT NULL,

PRIMARY KEY (caf\_id));

CREATE TABLE staff (

staff\_id INT NOT NULL,

staff\_name VARCHAR(45) NOT NULL,

caf\_id INT NOT NULL,

PRIMARY KEY (staff\_id),

FOREIGN KEY (caf\_id) REFERENCES cafedra(caf\_id) ON DELETE NO ACTION ON UPDATE NO ACTION);

ALTER TABLE cafedra ADD FOREIGN KEY (manager\_id) REFERENCES staff(staff\_id) ON DELETE NO ACTION ON UPDATE NO ACTION;

**2 Вариант**

CREATE TABLE staff(

staff\_id INT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

caf\_id INT,

FOREIGN KEY (caf\_id) REFERENCES cafedra (caf\_id) ON DELETE

SET

NULL

);

CREATE TABLE cafedra(

caf\_id INT NOT NULL,

name VARCHAR(50) NOT NULL UNIQUE,

PRIMARY KEY (caf\_id),

FOREIGN KEY (caf\_id) REFERENCES staff (staff\_id) ON DELETE NO ACTION

);

## 3.Из той же оперы:кафедры и факультеты.

кафедры входят в состав факультета(то есть у факультета много кафедр, но как минимум есть одна всегда), а кафедра входит в состав только одного факультета

create table faculty(

fac\_id INT,

neccesery\_caf INT NOT NULL,

PRIMARY KEY(fac\_id),

`name` VARCHAR(45) NOT NULL,

FOREIGN KEY(nececcery\_caf) REFERENCES cafedra(caf\_id)

);

create table cafedra(

caf\_id INT,

faculty\_id INT,

PRIMARY KEY(caf\_id),

FOREIGN KEY(faculty\_id) REFERENCES faculty(fac\_id)

);

## 

## 

## 

## 

## 

## 4. Можно ли использовать в реляционных бд тип данных, у которого не введены операторы = и ==? Я ответила, что теоретически ввести можно, но практической пользы от этого не будет

Про == сказала, что тогда ограничение домена не должно содержать операции сравнения на равенство и что при операциях выборки, например, тоже логическая функция должна быть без сравнения на равенство

Ну и что в таком случае функционал сокращается и на практике в этом нет особой пользы

**Из фундаментальных свойств реляционных понятий следует, что В теле любого отношения все кортежи различаются. следовательно == нельзя не определить**

А про = сказала, что в таком случае мы вообще не сможем использовать присваивание, тогда не получится вообще заполнять данные и тогда смысла в этом тоже нет

В нюансах всяких не уверена, но видимо он всё-таки принял этот ответ

**мы можем = переопределить через > < и не**

## 5. Доп был для 20 рандомных целых чисел, рассматривая числа как значение некоторого атрибута отношения

## проиндексировать отношение с использованием древовидного индекса на основе B+-дерева со степенью 2 и ещё каким-то ограничением с индексами

Сайт с построением B+-дерева онлайн: <https://www.cs.usfca.edu/~galles/visualization/BTree.html>

## 

## 

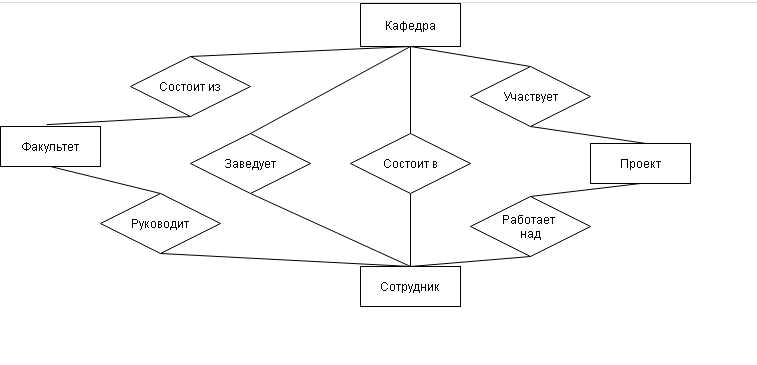
## 

## 

## 

## 6. Есть ER Диаграмма информационной системы(как он сказал, не совсем до конца оформленная) для учета структуры некоторого учебного заведения. В этом учебном заведении присутствуют кафедры, работники этого уч заведения, факультеты и проекты над которыми трудятся работники. Как в варианте языка определения данных, можно сформулировать требования, что сотрудники могут работать над какими-либо проектами, при этом над каким-либо проектом обязательно должен трудиться хотя бы один сотрудник?

(Ниже скинул диаграмму, которую он мне скидывал)



create table employee(

emp\_id INT,

name INT NOT NULL,

PRIMARY KEY(empl\_id),

);

create table emp\_to\_proj(

emp\_id INT NOT NULL,

proj\_id INT NOT NULL,

FOREIGN KEY(emp\_id) REFERENCES employee(emp\_id)

FOREIGN KEY(proj\_id) REFERENCES employee(proj\_id)

);

create table project(

proj\_id INT,

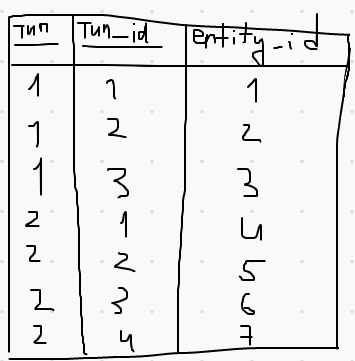
name INT NOT NULL,

PRIMARY KEY(proj\_id),

);

## 7. Вопрос, как в базе, которая sql сделать так, чтобы идентификаторы каждой таблицы были разные

Можно сделать такую таблицу



тип - это тип объекта (отрезок, дуга, ...), должен быть внешним ключом для objtype

тип\_id - это уникальный id для объекта конкретного типа , должен быть внешним ключом для object

entity\_id - это id в таблице entity, должен быть внешним ключом к entity. Там уже сквозная нумерация id, но в объектах разных типов конечно могут быть одни и те же id, например отрезок с id 8 и окружность с id 8 будут иметь разные id в этом столбце

entity\_id может быть null

**Можно изменить во всех таблицах тип id с инт на что-то большее целочисленное и прибавить ко всем таблицам большие разные числа ex если id от 0 до 1000 изменить тип на больший и добавить к id в каждой таблице разное число начиная от 1000000, 1200000 ...**

## 8. Есть отношение и 3 атрибута A,B,C, где A и B образуют потенциальный ключ, нужно составить функциональные зависимости чтобы получилась 3НФ, а не НФБК

По идее должен быть ещё какой-то потенциальный ключ, например {B, C} (чтобы пересечение с {A,B} было непусто). Тогда помимо зависимостей от потенциальных ключей ( {A ,B} C, {B, C} A ) можно добавить зависимость CA. Транзитивных нетривиальных зависимостей нет (появляется только {A ,B} A, это тривиальная), т.е. условия для 3нф выполняются. Зависимость CA не обладает потенциальным ключом в качестве детерминанта, значит условия для НФБК не выполняются.

Или если не добавлять потенциальный ключ, наверное можно и просто **CA**, но это хз.

Да, так тоже сработает, потому что тогда {B, C} автоматически станет потенциальным ключом.

## **9. Таблица с двумя столбцами и условием, что и значение и первого и второго атрибутов принадлежит какому-то множеству С. Вопрос: какие функциональные зависимости есть и в какой нормальной форме**

Атрибуты A,B. Если A- ключ, то A->B

Если B-ключ, то B->A

Здесь есть только две тривиальные зависимости AB->A, AB->B

**Нормальная форма четвёртая** - 1НФ у всех отношений, для остальных нужно рассмотреть все возможные потенциальные ключи (АВ, А, В) и понятно что неключевые атрибуты неприводимо только могут зависеть -2НФ, по той же логике не будет и транзитивных зависимостей - 3НФ, зависимости есть только тривиальные - НФБК, и многозначных нет, поэтому 4НФ

## 10.Есть бд с объектами, и ограничениями на эти объекты. Как сделать так, чтоб можно было создавать ограничения в зависимости от типа объекта и их количества.

Например: у 1-го отрезка можно создать ограничение длины, у 2-х угла между ними и тд

Связываем таблицу constraint и object. Это дает нам то, что в object для нескольких объектов мы можем делать ограничения, не обязательно для одного

Можно для справки(чтоб мы знали какие ограничения и сколько ограниченных объектов для каждого типа ограничения) создать третью таблицу info, которая будет содержать атрибуты type и number\_of\_object. type - первичный ключ, связан с type в constraint и внешний для constraint. number\_of\_object задаем сами

Если number\_of\_object != количеству вставляемых объектов, тогда нельзя вставить

(фото ниже)

## 

## 11. Построение нескольких табличек на языке реляционных БД: нужно построить БД для ситуации, когда сотрудник ВУЗа может работать в одиночку или на кафедре.(точно верное решение)

CREATE DATABASE IF NOT EXISTS university;

use university;

CREATE TABLE worker (

worker\_id INT,

name VARCHAR(30),

PRIMARY KEY (worker\_id)

);

CREATE TABLE cathedra (

catherdra\_id INT,

name VARCHAR(30),

PRIMARY KEY (catherdra\_id)

);

CREATE TABLE worker\_of\_cathedra (

catherdra\_id INT,

worker\_id INT,

PRIMARY KEY (worker\_id),

FOREIGN KEY (worker\_id)

REFERENCES worker(worker\_id),

FOREIGN KEY (cathedra\_id)

REFERENCES cathedra(cathedra\_id)

);

## 

## 

## 

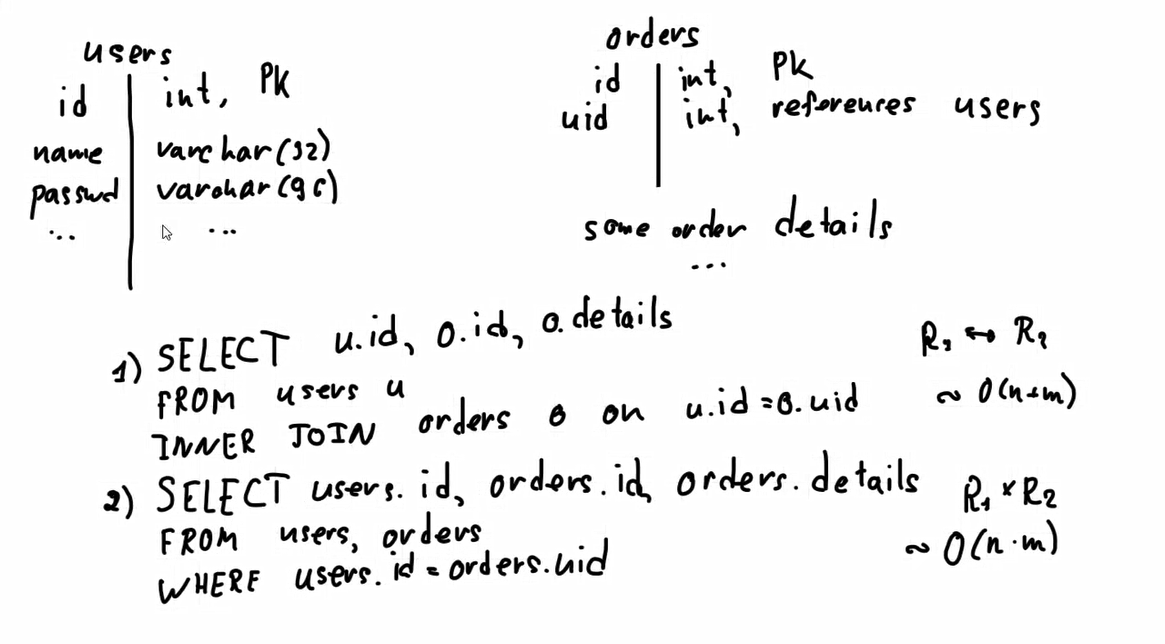
## 

## 12. Изучить схему базы, по которой давал SQL запрос

И спросил что нужно, чтобы точки чертежа все находились на одном уровне

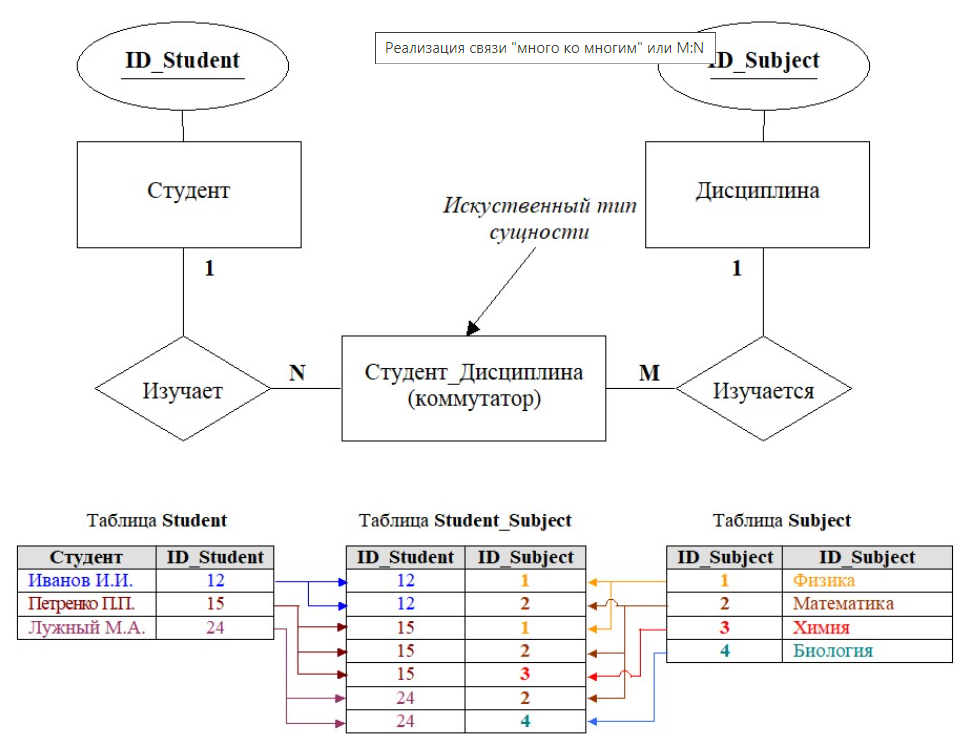
## 

## 13. Привести два эквивалентных SQL-запроса, которые бы требовали сильно разное число операций



## 14. Перевод ER-модели и реализацию обязательной связи в отношении многие-ко-многим

При реализации связи многие-ко-многим, допустимой в инфологической модели, производится ее преобразование к связям один-ко-многим, например, через промежуточное отношение (рисунок 11.3 – отношение «Связь»). Промежуточное отношение будет иметь первичный ключ, состоящий из первичных ключей связываемых отношений.

Для необязательных типов связи на физическом уровне у атрибутов, соответствующих внешнему ключу, устанавливается свойство допустимости неопределенных значений. При обязательном типе связи атрибуты получают свойство отсутствия неопределенных значений.  
  


## 15. Есть отношение с атрибутами A, B, C, D

При этом A->B, C->D (функц. зависимость), больше ничего неизвестно про эти атрибуты и отношение. Можно ли провести декомпозицию этого отношения без потерь на 2 отношения с атрибутами A, B и C, D?

**Ответ:** Исходя из теоремы Хита нельзя потому что у этих отношений нет общих атрибутов, из-за чего операция естественного слияния сфейлится. Вроде правильно

## 16. Как представить xml документ в виде таблиц

Если не совсем строго говорить, то с заданием таблиц со связями 1 к 1 или 1 к многим (используется внешний ключ), при переходе на следующий уровень вложенности - используется новая таблица

## 

## 

## 17. Отношение содержит 6 атрибутов А, В, С, D, E, F фз: A -> {B,C}, C->A, D -> E, F->A, E->D Будет ли разбиение отношения на ACD BCD EFD декомпозицией без потери информации?

Вроде можно, если A, {B,C} и F - первичные + ФЗ: С->A ; E->D сохраняются в декомпозированных отношениях