

Теперь можно познакомиться с данными, которые хранят таблицы.

acquisition

Содержит информацию о покупках одних компаний другими.

Таблица включает такие поля:

* первичный ключ id — идентификатор или уникальный номер покупки;
* внешний ключ acquiring\_company\_id — ссылается на таблицу company — идентификатор компании-покупателя, то есть той, что покупает другую компанию;
* внешний ключ acquired\_company\_id — ссылается на таблицу company — идентификатор компании, которую покупают;
* term\_code — способ оплаты сделки:
  + cash — наличными;
  + stock — акциями компании;
  + cash\_and\_stock — смешанный тип оплаты: наличные и акции.
* price\_amount — сумма покупки в долларах;
* acquired\_at — дата совершения сделки;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

company

Содержит информацию о компаниях-стартапах.

* первичный ключ id — идентификатор, или уникальный номер компании;
* name — название компании;
* category\_code — категория деятельности компании, например:
  + news — специализируется на работе с новостями;
  + social — специализируется на социальной работе.
* status — статус компании:
  + acquired — приобретена;
  + operating — действует;
  + ipo — вышла на IPO;
  + closed — перестала существовать.
* founded\_at — дата основания компании;
* closed\_at — дата закрытия компании, которую указывают в том случае, если компании больше не существует;
* domain — домен сайта компании;
* twitter\_username — название профиля компании в твиттере;
* country\_code — код страны, например, USA для США, GBR для Великобритании;
* investment\_rounds — число раундов, в которых компания участвовала как инвестор;
* funding\_rounds — число раундов, в которых компания привлекала инвестиции;
* funding\_total — сумма привлечённых инвестиций в долларах;
* milestones — количество важных этапов в истории компании;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

education

Хранит информацию об уровне образования сотрудников компаний.

* первичный ключ id — уникальный номер записи с информацией об образовании;
* внешний ключ person\_id — ссылается на таблицу people — идентификатор человека, информация о котором представлена в записи;
* degree\_type — учебная степень, например:
  + BA — Bachelor of Arts — бакалавр гуманитарных наук;
  + MS — Master of Science — магистр естественных наук.
* instituition — учебное заведение, название университета;
* graduated\_at — дата завершения обучения, выпуска;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

fund

Хранит информацию о венчурных фондах.

* первичный ключ id — уникальный номер венчурного фонда;
* name — название венчурного фонда;
* founded\_at — дата основания фонда;
* domain — домен сайта фонда;
* twitter\_username — профиль фонда в твиттере;
* country\_code — код страны фонда;
* investment\_rounds — число инвестиционных раундов, в которых фонд принимал участие;
* invested\_companies — число компаний, в которые инвестировал фонд;
* milestones — количество важных этапов в истории фонда;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

funding\_round

Содержит информацию о раундах инвестиций.

* первичный ключ id — уникальный номер инвестиционного раунда;
* внешний ключ company\_id — ссылается на таблицу company — уникальный номер компании, участвовавшей в инвестиционном раунде;
* funded\_at — дата проведения раунда;
* funding\_round\_type — тип инвестиционного раунда, например:
  + venture — венчурный раунд;
  + angel — ангельский раунд;
  + series\_a — раунд А.
* raised\_amount — сумма инвестиций, которую привлекла компания в этом раунде в долларах;
* pre\_money\_valuation — предварительная, проведённая до инвестиций оценка стоимости компании в долларах;
* participants — количество участников инвестиционного раунда;
* is\_first\_round — является ли этот раунд первым для компании;
* is\_last\_round — является ли этот раунд последним для компании;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

investment

Содержит информацию об инвестициях венчурных фондов в компании-стартапы.

* первичный ключ id — уникальный номер инвестиции;
* внешний ключ funding\_round\_id — ссылается на таблицу funding\_round — уникальный номер раунда инвестиции;
* внешний ключ company\_id — ссылается на таблицу company — уникальный номер компании-стартапа, в которую инвестируют;
* внешний ключ fund\_id — ссылается на таблицу fund — уникальный номер фонда, инвестирующего в компанию-стартап;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

people

Содержит информацию о сотрудниках компаний-стартапов.

* первичный ключ id — уникальный номер сотрудника;
* first\_name — имя сотрудника;
* last\_name — фамилия сотрудника;
* внешний ключ company\_id — ссылается на таблицу company — уникальный номер компании-стартапа;
* twitter\_username — профиль сотрудника в твиттере;
* created\_at — дата и время создания записи в таблице;
* updated\_at — дата и время обновления записи в таблице.

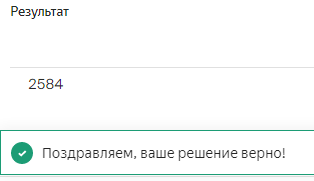
1.

Посчитайте, сколько компаний закрылось.

SELECT count(\*)

FROM company

WHERE status = 'closed'



2.

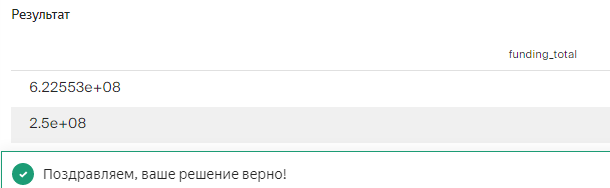
Отобразите количество привлечённых средств для новостных компаний США. Используйте данные из таблицы company. Отсортируйте таблицу по убыванию значений в поле funding\_total .

SELECT funding\_total

FROM company

WHERE category\_code = 'news' AND country\_code = 'USA'

ORDER BY funding\_total DESC



3.

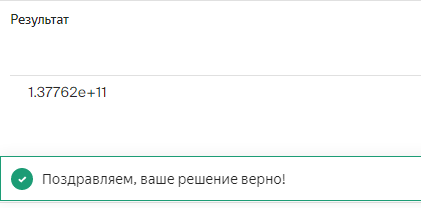
Найдите общую сумму сделок по покупке одних компаний другими в долларах. Отберите сделки, которые осуществлялись только за наличные с 2011 по 2013 год включительно.

SELECT SUM(price\_amount)

FROM acquisition

WHERE term\_code='cash' AND

acquired\_at BETWEEN '2011-01-01' AND '2013-12-31'



4.

Отобразите имя, фамилию и названия аккаунтов людей в твиттере, у которых названия аккаунтов начинаются на 'Silver'.

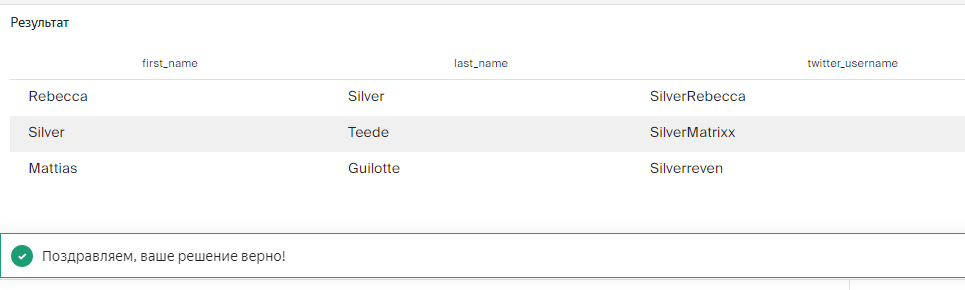
SELECT first\_name,

last\_name,

twitter\_username

FROM people

WHERE twitter\_username LIKE 'Silver%'



5.

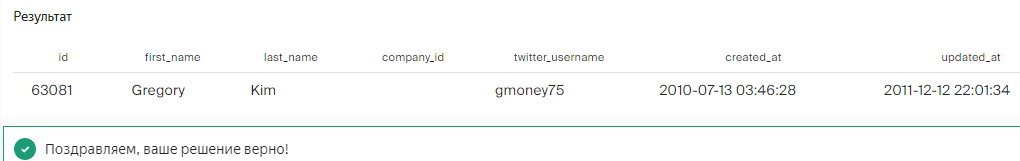
Выведите на экран всю информацию о людях, у которых названия аккаунтов в твиттере содержат подстроку 'money', а фамилия начинается на 'K'.

SELECT \*

FROM people

WHERE twitter\_username LIKE '%money%' AND

last\_name LIKE 'K%'



6.

Для каждой страны отобразите общую сумму привлечённых инвестиций, которые получили компании, зарегистрированные в этой стране. Страну, в которой зарегистрирована компания, можно определить по коду страны. Отсортируйте данные по убыванию суммы.

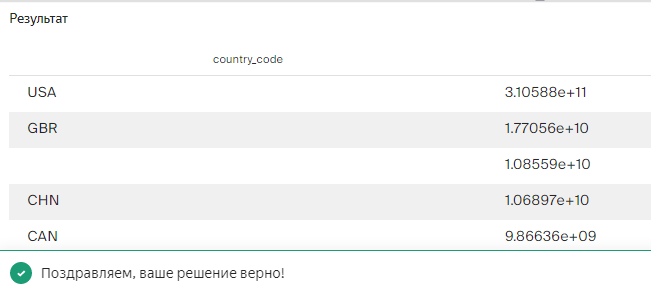
SELECT country\_code,

SUM(funding\_total) as total

FROM company

GROUP BY country\_code

ORDER BY total DESC



7.

Составьте таблицу, в которую войдёт дата проведения раунда, а также минимальное и максимальное значения суммы инвестиций, привлечённых в эту дату.

Оставьте в итоговой таблице только те записи, в которых минимальное значение суммы инвестиций не равно нулю и не равно максимальному значению.

WITH sql1 AS (SELECT funded\_at as dt,

MIN(raised\_amount) as mn,

MAX(raised\_amount) as mx

FROM funding\_round

GROUP BY funded\_at

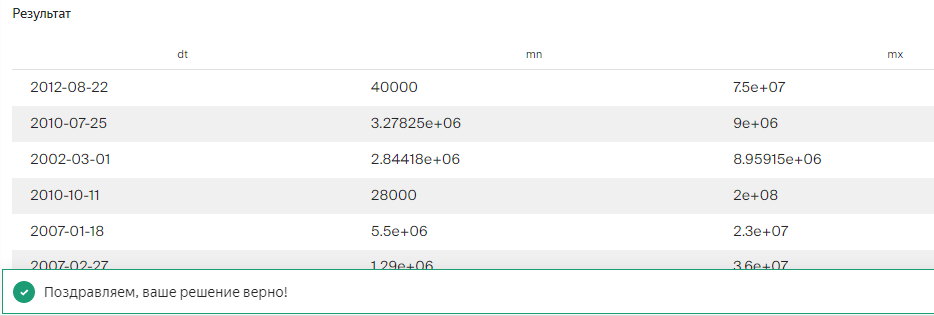
)

SELECT \*

FROM sql1

WHERE mn!=0 AND

mn!=mx



8.

Создайте поле с категориями:

* Для фондов, которые инвестируют в 100 и более компаний, назначьте категорию high\_activity.
* Для фондов, которые инвестируют в 20 и более компаний до 100, назначьте категорию middle\_activity.
* Если количество инвестируемых компаний фонда не достигает 20, назначьте категорию low\_activity.

Отобразите все поля таблицы fund и новое поле с категориями.

SELECT \*,

CASE

WHEN invested\_companies <20 THEN 'low\_activity'

WHEN invested\_companies >=20 AND invested\_companies <100 THEN 'middle\_activity'

ELSE 'high\_activity'

END

FROM fund;

9.

Для каждой из категорий, назначенных в предыдущем задании, посчитайте округлённое до ближайшего целого числа среднее количество инвестиционных раундов, в которых фонд принимал участие. Выведите на экран категории и среднее число инвестиционных раундов. Отсортируйте таблицу по возрастанию среднего.

WITH sql1 AS (SELECT \*,

CASE

WHEN invested\_companies>=100 THEN 'high\_activity'

WHEN invested\_companies>=20 THEN 'middle\_activity'

ELSE 'low\_activity'

END AS activity

FROM fund

)

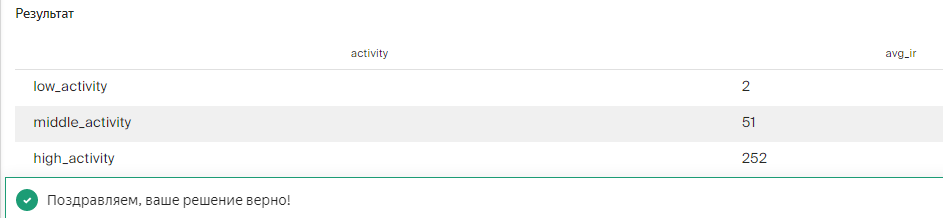
SELECT activity,

ROUND(AVG(investment\_rounds)) as avg\_ir

FROM sql1

GROUP BY activity

ORDER BY avg\_ir



10.

Проанализируйте, в каких странах находятся фонды, которые чаще всего инвестируют в стартапы.

Для каждой страны посчитайте минимальное, максимальное и среднее число компаний, в которые инвестировали фонды этой страны, основанные с 2010 по 2012 год включительно. Исключите страны с фондами, у которых минимальное число компаний, получивших инвестиции, равно нулю.

Выгрузите десять самых активных стран-инвесторов: отсортируйте таблицу по среднему количеству компаний от большего к меньшему. Затем добавьте сортировку по коду страны в лексикографическом порядке.

SELECT country\_code as cc,

MIN(invested\_companies) as mn,

MAX(invested\_companies) as mx,

AVG(invested\_companies) as av

FROM fund

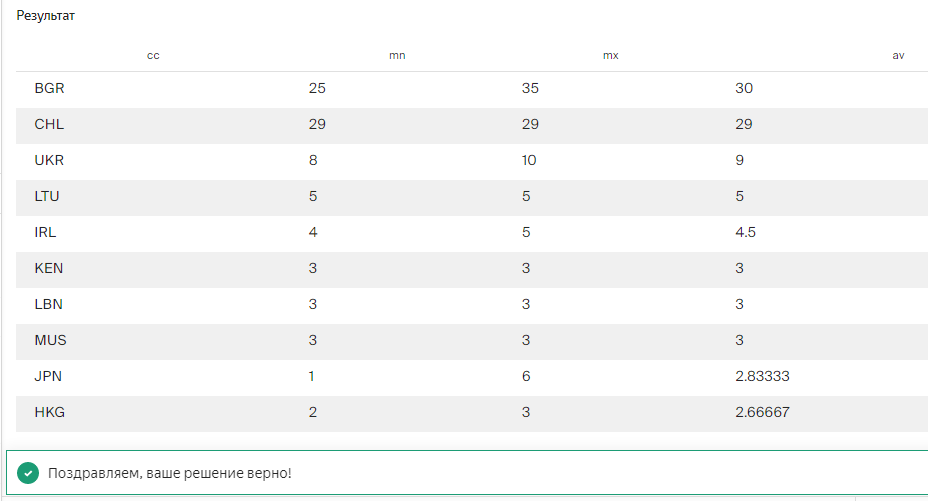
WHERE founded\_at BETWEEN '2010-01-01' AND '2012-12-31'

GROUP BY cc

HAVING MIN(invested\_companies)!=0

ORDER BY av DESC, cc ASC

LIMIT 10



11.

Отобразите имя и фамилию всех сотрудников стартапов. Добавьте поле с названием учебного заведения, которое окончил сотрудник, если эта информация известна.

SELECT p.first\_name,

p.last\_name,

e.instituition

FROM people as p

LEFT JOIN education as e

ON p.id = e.person\_id

12.

Для каждой компании найдите количество учебных заведений, которые окончили её сотрудники. Выведите название компании и число уникальных названий учебных заведений. Составьте топ-5 компаний по количеству университетов.

WITH sql1 AS (SELECT c.name as cn,

e.instituition as ins

FROM company as c

INNER JOIN

people as p ON c.id = p.company\_id

INNER JOIN

education as e ON p.id = e.person\_id

GROUP BY cn,ins

)

SELECT cn,

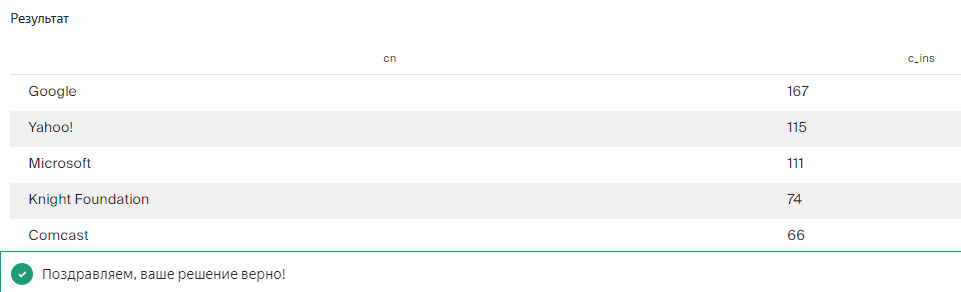
COUNT(ins) as c\_ins

FROM sql1

GROUP BY cn

ORDER BY c\_ins DESC

LIMIT 5



13.

Составьте список с уникальными названиями закрытых компаний, для которых первый раунд финансирования оказался последним.

SELECT DISTINCT name

FROM company as c

INNER JOIN

(SELECT company\_id as ci,

is\_first\_round,

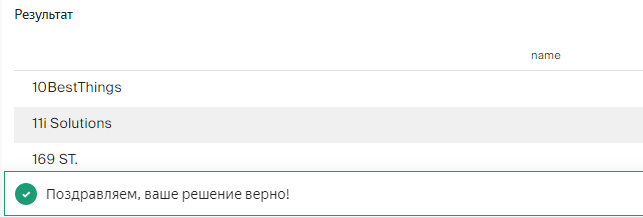
is\_last\_round

FROM funding\_round

WHERE is\_first\_round=1 AND is\_last\_round=1) as sql1 ON

c.id = sql1.ci

WHERE status = 'closed'



14.

Составьте список уникальных номеров сотрудников, которые работают в компаниях, отобранных в предыдущем задании.

WITH sql2 AS (

SELECT DISTINCT id

FROM company as c

INNER JOIN

(SELECT company\_id as ci,

is\_first\_round,

is\_last\_round

FROM funding\_round

WHERE is\_first\_round=1 AND is\_last\_round=1) as sql1 ON

c.id = sql1.ci

WHERE status = 'closed'

)

SELECT DISTINCT p.id

FROM people as p

INNER JOIN

sql2 ON sql2.id = p.company\_id

15.

Составьте таблицу, куда войдут уникальные пары с номерами сотрудников из предыдущей задачи и учебным заведением, которое окончил сотрудник.

WITH sql2 AS (

SELECT DISTINCT id

FROM company as c

INNER JOIN

(SELECT company\_id as ci,

is\_first\_round,

is\_last\_round

FROM funding\_round

WHERE is\_first\_round=1 AND is\_last\_round=1) as sql1 ON

c.id = sql1.ci

WHERE status = 'closed'

)

SELECT DISTINCT p.id, e.instituition

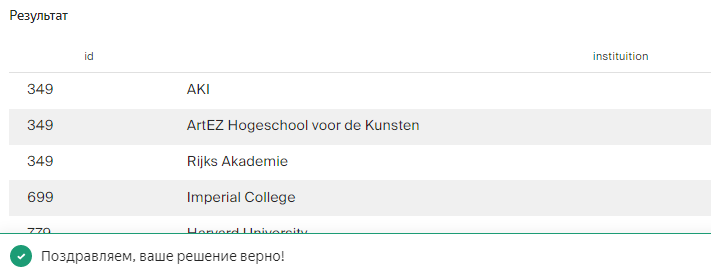
FROM people as p

INNER JOIN

sql2 ON sql2.id = p.company\_id

INNER JOIN

education as e ON p.id = e.person\_id



16.

Посчитайте количество учебных заведений для каждого сотрудника из предыдущего задания. При подсчёте учитывайте, что некоторые сотрудники могли окончить одно и то же заведение дважды.

WITH sql2 AS (

SELECT DISTINCT id

FROM company as c

INNER JOIN

(SELECT company\_id as ci,

is\_first\_round,

is\_last\_round

FROM funding\_round

WHERE is\_first\_round=1 AND is\_last\_round=1) as sql1 ON

c.id = sql1.ci

WHERE status = 'closed'

),

sql3 AS (SELECT p.id as pers\_id,

e.instituition as pers\_inst

FROM people as p

INNER JOIN

sql2 ON sql2.id = p.company\_id

INNER JOIN

education as e ON p.id = e.person\_id

)

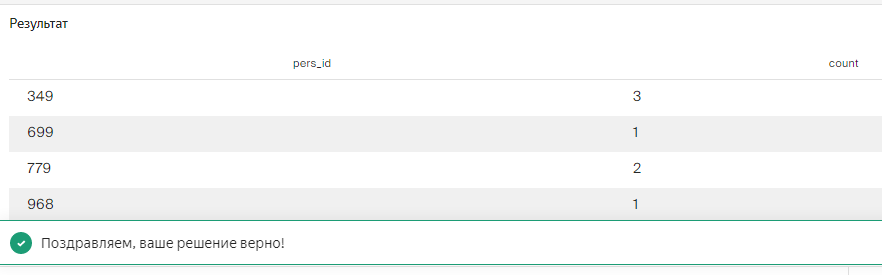
Select distinct pers\_id,

COUNT(\*) OVER (PARTITION BY pers\_id)

FROM sql3

ORDER BY pers\_id

!!! УБРАН DISTINCT из sql3



17.

Дополните предыдущий запрос и выведите среднее число учебных заведений (всех, не только уникальных), которые окончили сотрудники разных компаний. Нужно вывести только одну запись, группировка здесь не понадобится.

WITH sql2 AS (

SELECT DISTINCT id

FROM company as c

INNER JOIN

(SELECT company\_id as ci,

is\_first\_round,

is\_last\_round

FROM funding\_round

WHERE is\_first\_round=1 AND is\_last\_round=1) as sql1 ON

c.id = sql1.ci

WHERE status = 'closed'

),

sql3 AS (SELECT p.id as pers\_id,

e.instituition as pers\_inst

FROM people as p

INNER JOIN

sql2 ON sql2.id = p.company\_id

INNER JOIN

education as e ON p.id = e.person\_id

),

sql4 AS (Select DISTINCT pers\_id,

COUNT(\*) OVER (PARTITION BY pers\_id) as count\_inst

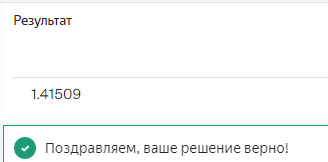
FROM sql3

ORDER BY pers\_id

)

SELECT AVG(count\_inst)

from sql4



18.

Напишите похожий запрос: выведите среднее число учебных заведений (всех, не только уникальных), которые окончили сотрудники Facebook\*.

\*(сервис, запрещённый на территории РФ)

WITH sql1 AS (

SELECT c.id as c\_id

FROM company as c

WHERE name = 'Facebook'

),

-- Отбор сотрудников, работающих в компаниях, id которых найден в sql1

sql2 AS (

SELECT p.id as p\_id

FROM people as p

INNER JOIN sql1 ON

p.company\_id = sql1.c\_id

),

sql3 AS (

SELECT p\_id,

e.instituition,

e.graduated\_at,

e.created\_at

FROM sql2

INNER JOIN

education as e ON

sql2.p\_id = e.person\_id

--WHERE e.graduated\_at IS NOT NULL

),

sql4 AS (

SELECT DISTINCT p\_id,

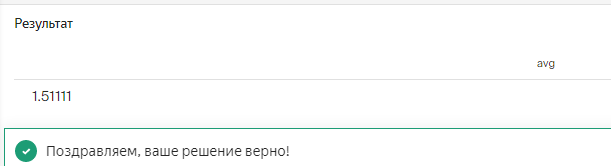
COUNT(\*) OVER (PARTITION BY p\_id) as count\_inst

FROM sql3

)

SELECT AVG(count\_inst)

FROM sql4



19.

Составьте таблицу из полей:

* name\_of\_fund — название фонда;
* name\_of\_company — название компании;
* amount — сумма инвестиций, которую привлекла компания в раунде.

В таблицу войдут данные о компаниях, в истории которых было больше шести важных этапов, а раунды финансирования проходили с 2012 по 2013 год включительно.

WITH

sql1 AS (

SELECT c.id as comp\_id,

c.name as comp\_name,

fr.id as fr\_id,

fr.raised\_amount as amount

FROM company as c

INNER JOIN

funding\_round as fr ON

c.id = fr.company\_id

WHERE fr.funded\_at BETWEEN '2012-01-01' AND '2013-12-31' AND

c.milestones > 6

),

sql2 AS (

SELECT f.name as name\_of\_fund,

sql1.comp\_name AS name\_of\_company,

sql1.amount as amount

FROM investment as i

INNER JOIN

sql1 ON i.funding\_round\_id = sql1.fr\_id

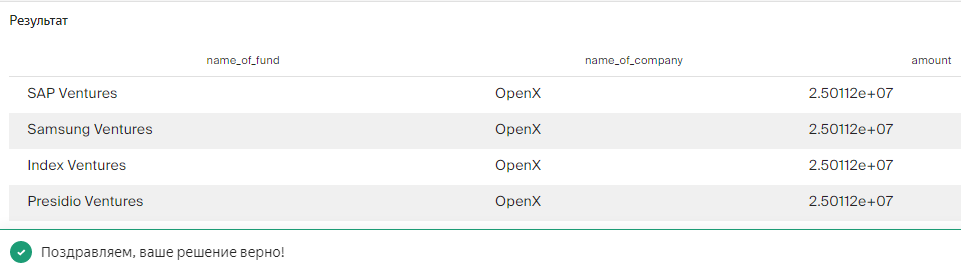
INNER JOIN

fund as f ON i.fund\_id = f.id

)

SELECT \*

FROM sql2



20.

Выгрузите таблицу, в которой будут такие поля:

* название компании-покупателя;
* сумма сделки;
* название компании, которую купили;
* сумма инвестиций, вложенных в купленную компанию;
* доля, которая отображает, во сколько раз сумма покупки превысила сумму вложенных в компанию инвестиций, округлённая до ближайшего целого числа.

Не учитывайте те сделки, в которых сумма покупки равна нулю. Если сумма инвестиций в компанию равна нулю, исключите такую компанию из таблицы.

Отсортируйте таблицу по сумме сделки от большей к меньшей, а затем по названию купленной компании в лексикографическом порядке. Ограничьте таблицу первыми десятью записями.

SELECT

c1.name as n1,

a.price\_amount,

c2.name as n2,

c2.funding\_total,

ROUND(a.price\_amount/c2.funding\_total) as part\_pr

FROM acquisition as a

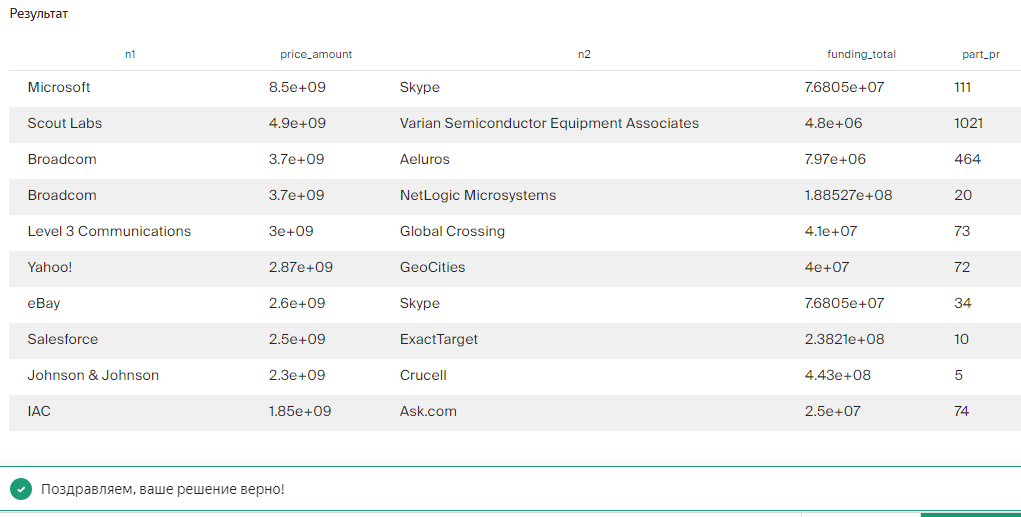
INNER JOIN company as c1 ON a.acquiring\_company\_id = c1.id

INNER JOIN company as c2 ON a.acquired\_company\_id = c2.id

WHERE a.price\_amount !=0 AND c2.funding\_total != 0

ORDER BY a.price\_amount DESC, n2 ASC

LIMIT 10



21.

Выгрузите таблицу, в которую войдут названия компаний из категории social, получившие финансирование с 2010 по 2013 год включительно. Проверьте, что сумма инвестиций не равна нулю. Выведите также номер месяца, в котором проходил раунд финансирования.

SELECT c.name,

--fr.raised\_amount,

--DATE\_TRUNC('month',fr.funded\_at)::date month\_raund

EXTRACT('MONTH' from fr.funded\_at) as month\_raund

FROM company as c

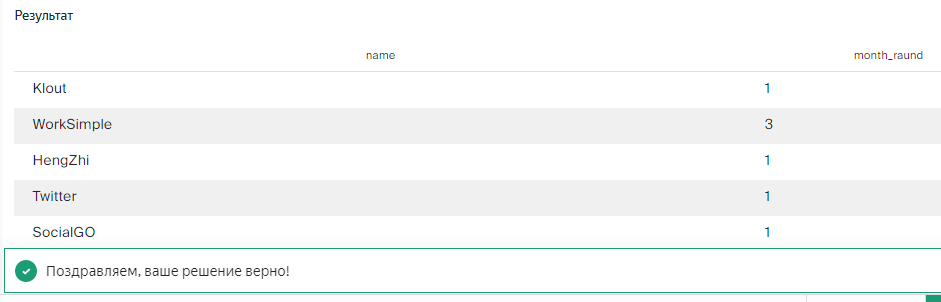
INNER JOIN

funding\_round as fr ON c.id = fr.company\_id

WHERE c.category\_code = 'social' AND

fr.funded\_at BETWEEN '2010-01-01' AND '2013-12-31' AND

fr.raised\_amount!=0



22.

Отберите данные по месяцам с 2010 по 2013 год, когда проходили инвестиционные раунды. Сгруппируйте данные по номеру месяца и получите таблицу, в которой будут поля:

* номер месяца, в котором проходили раунды;
* количество уникальных названий фондов из США, которые инвестировали в этом месяце;
* количество компаний, купленных за этот месяц;
* общая сумма сделок по покупкам в этом месяце.

WITH

sql1 AS (

SELECT id as fund\_id

FROM fund

WHERE country\_code = 'USA'

),

sql2 AS (

SELECT

EXTRACT('MONTH' from funded\_at) as f\_at,

i.fund\_id as f\_id

FROM funding\_round as fr

INNER JOIN

investment as i ON fr.id = i.funding\_round\_id

INNER JOIN sql1 ON i.fund\_id = sql1.fund\_id

WHERE fr.funded\_at BETWEEN '2010-01-01' AND '2013-12-31'

),

sql3 AS (

SELECT

EXTRACT('MONTH' from acquired\_at) as ac\_at,

COUNT(acquired\_company\_id) as c\_pay,

SUM(price\_amount) as r\_am

FROM acquisition

WHERE acquired\_at BETWEEN '2010-01-01' AND '2013-12-31'

GROUP BY ac\_at

)

SELECT f\_at,

count (distinct f\_id),

c\_pay,

r\_am

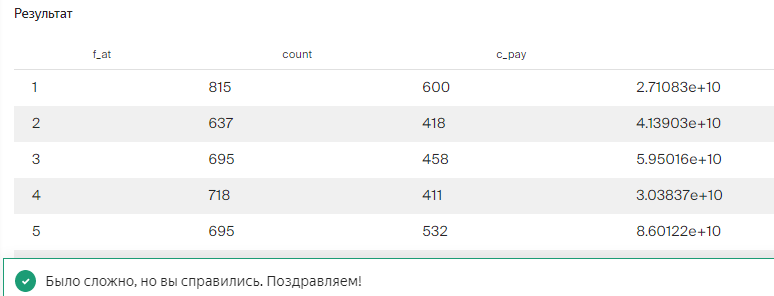
FROM sql2

LEFT JOIN

sql3 ON sql2.f\_at = sql3.ac\_at

GROUP BY f\_at,c\_pay,r\_am

ORDER BY f\_at



23.

Составьте сводную таблицу и выведите среднюю сумму инвестиций для стран, в которых есть стартапы, зарегистрированные в 2011, 2012 и 2013 годах. Данные за каждый год должны быть в отдельном поле. Отсортируйте таблицу по среднему значению инвестиций за 2011 год от большего к меньшему.

WITH

sql2011 AS (

SELECT

country\_code as cc\_2011,

AVG(funding\_total) as avg\_2011

FROM company

WHERE EXTRACT('YEAR' FROM CAST(founded\_at as date)) = 2011

group by country\_code

),

sql2012 AS (

SELECT

country\_code as cc\_2012,

AVG(funding\_total) as avg\_2012

FROM company

WHERE EXTRACT('YEAR' FROM CAST(founded\_at as date)) = 2012

group by country\_code

),

sql2013 AS (

SELECT

country\_code as cc\_2013,

AVG(funding\_total) as avg\_2013

FROM company

WHERE EXTRACT('YEAR' FROM CAST(founded\_at as date)) = 2013

group by country\_code

)

SELECT

cc\_2011 AS c\_code,

avg\_2011 as av\_2011,

avg\_2012 as av\_2012,

avg\_2013 as av\_2013

FROM

sql2011

INNER JOIN sql2012 ON sql2011.cc\_2011 = sql2012.cc\_2012

INNER JOIN sql2013 ON sql2011.cc\_2011 = sql2013.cc\_2013

ORDER BY

av\_2011 DESC

