Решение задач по SQL

Сергей Заякин, май 2022 г.

ЗАДАЧА 1

Нужно посмотреть продажи по годам. Выгрузите таблицу, в которую войдут:

- год покупки;
- минимальная сумма заказа;
- максимальная сумма заказа;
- общая сумма выручки;
- количество заказов;
- средняя выручка на уникального покупателя, округлённая до ближайшего целого числа.

Отсортируйте таблицу по году от большего к меньшему. Отберите только те записи, в которых в поле `billing_country` указаны страны: США (англ. USA), Великобритания (англ. United Kingdom) и Германия (англ. Germany).

```
SELECT EXTRACT(YEAR FROM CAST(invoice_date AS date)),
       MIN(total),
       MAX(total),
       SUM(total),
       COUNT(invoice_id),
       ROUND(SUM(total)/COUNT(DISTINCT customer_id))
FROM invoice
WHERE billing_country IN ('USA', 'United Kingdom', 'Germany')
GROUP BY EXTRACT(YEAR FROM CAST(invoice_date AS date))
ORDER BY EXTRACT(YEAR FROM CAST(invoice_date AS date)) DESC
;
```

ЗАДАЧА 2

Отберите пятерых самых активных клиентов в США с 25 мая 2011 по 25 сентября 2011. Дату хранит поле `invoice_date`
, тип данных поля — `varchar`

```
SELECT customer_id,
       COUNT(customer_id)
FROM invoice
WHERE CAST(invoice_date AS date) BETWEEN '2011-05-25' AND '2011-09-25'
      AND billing_country = 'USA'-- сюда впишите условия
GROUP BY customer_id
ORDER BY COUNT(customer_id) DESC-- добавьте данные для сортировки
LIMIT 5;
```

ЗАДАЧА 3

Создайте новое поле с категориями:

- заказы на сумму меньше одного доллара получат категорию `'low cost'`;
- заказы на сумму от одного доллара и выше получат категорию `'high cost'`.

Для каждой категории посчитайте сумму значений в поле `total`, но только для тех заказов, при оформлении которых указан почтовый код. В итоговую таблицу должны войти только два поля.

```
SELECT SUM(total),
      CASE
          WHEN total < 1 THEN 'low cost'
          WHEN total >= 1 THEN 'high cost'
      END
FROM invoice
WHERE billing_postal_code IS NOT NULL
GROUP BY
      CASE
          WHEN total < 1 THEN 'low cost'
          WHEN total >= 1 THEN 'high cost'
      END
;
```

ЗАДАЧА 4

Сравните фильмы разных возрастных рейтингов. Найдите среднее значение цены аренды фильма в поле `rental_rate` для каждого рейтинга (поле `rating`). Оставьте в таблице только те записи, в которых среднее значение `rental_rate`

больше `3`

.

```
SELECT AVG(rental_rate),
       rating -- укажите нужные поля
FROM movie
GROUP BY rating-- сгруппируйте данные
HAVING AVG(rental_rate) > 3-- пропишите условие;
```

ЗАДАЧА 5

Изучите заказы, которые оформили в сентябре 2011 года. Сравните общую сумму выручки (поле `total`) за каждый день этого месяца: выведите день в формате `'2011-09-01'` и сумму. Информацию о дате заказа хранит поле `invoice_date`. Не забудьте изменить тип данных в этом поле, чтобы использовать операторы для работы с датой. Оставьте в таблице только те значения суммы, которые больше `1` и меньше `10`.

```
SELECT CAST(invoice_date AS date),
       SUM(total)-- укажите нужные поля
FROM invoice
WHERE CAST(invoice_date AS date) BETWEEN '2011-09-01' AND '2011-09-30' -- пропишите условие
GROUP BY CAST(invoice_date AS date)-- сгруппируйте данные
HAVING SUM(total) > 1 AND SUM(total) < 10-- добавьте условие;
```

ЗАДАЧА 6

Посчитайте пропуски в поле с почтовым индексом `billing_postal_code` для каждой страны (поле `billing_country`). Получите срез: в таблицу должны войти только те записи, в которых поле `billing_address` содержит слова `Street`, `Way`, `Road` или `Drive`. Отобразите в таблице страну и число пропусков, если их больше `6`.

```
SELECT billing_country,
       COUNT(*)
FROM invoice
WHERE (billing_address LIKE '%Street%'
      OR billing_address LIKE '%Way%'
      OR billing_address LIKE '%Road%'
      OR billing_address LIKE '%Drive%')
      AND billing_postal_code IS NULL-- пропишите условие
GROUP BY billing_country-- сгруппируйте данные
HAVING COUNT(billing_postal_code IS NULL) > 6-- пропишите условие;
```

ЗАДАЧА 7

Нужно объединить данные двух таблиц: `track` и `invoice_line`. Таблица `track` хранит информацию о музыкальных треках в магазине, названия треков указаны в поле `name`. Таблица `invoice_line` содержит данные о купленных треках, их стоимость указана в поле `unit_price`. В обеих таблицах есть поле `track_id` — в нём содержатся идентификаторы музыкальных треков.

Выгрузите таблицу, в которой названию трека будет соответствовать его стоимость. Отберите все уникальные записи. Если какой-либо из треков не покупали или у купленного трека нет названия — такие записи не должны войти в таблицу. Оставьте в итоговой таблице первые 20 записей.

```
SELECT DISTINCT t.name,
     i.unit_price
FROM track AS t
INNER JOIN invoice_line AS i ON t.track_id=i.track_id
WHERE  quantity != 0
       AND name IS NOT NULL
LIMIT 20
;
```

ЗАДАЧА 8

Нужно дополнить запрос: добавьте поле с идентификатором плейлиста `playlist_id`. Такое поле можно получить из таблицы `playlist_track`. В этой таблице собраны идентификаторы плейлистов и треков (поле `track_id`). Условие остаётся прежним: если идентификаторы треков не совпадают во всех трёх таблицах, такие треки не должны войти в итоговую таблицу. Выведите первые 20 записей.

```
SELECT t.name,
       i.unit_price,
       p.playlist_id-- укажите необходимое поле
FROM track AS t
INNER JOIN invoice_line AS i ON t.track_id=i.track_id
INNER JOIN playlist_track AS p ON t.track_id=P.track_id
LIMIT 20
-- добавьте ещё одно присоединение;
```

```
SELECT t.name,
i.unit_price,
pt.playlist_id,
p.name -- укажите необходимое поле
```

```
FROM track AS t
INNER JOIN invoice_line AS i ON t.track_id=i.track_id
INNER JOIN playlist_track AS pt ON t.track_id=pt.track_id
INNER JOIN playlist AS p ON pt.playlist_id = p.playlist_id
LIMIT 20
```

```
SELECT p.name AS playlist_name,
       SUM( i.unit_price) AS total_revenue
-- укажите нужные поля
FROM track AS t
INNER JOIN invoice_line AS i ON t.track_id=i.track_id
INNER JOIN playlist_track AS pl ON t.track_id=pl.track_id
INNER JOIN playlist AS p ON pl.playlist_id = p.playlist_id
GROUP BY playlist_name-- добавьте поле для группировки
ORDER BY total_revenue DESC-- добавьте поле для сортировки;
```

ЗАДАЧА 9

Сгруппируйте данные по жанрам и посчитайте количество заказов. Выведите на экран два поля: одно с названием жанра, второе — с количеством купленных треков в этом жанре. Отсортируйте таблицу по убыванию количества заказов.

```
SELECT g.name,
       COUNT(invoice_line_id) AS tracks_purchased
-- укажите нужные поля
FROM invoice AS iv
INNER JOIN invoice_line AS i ON iv.invoice_id=i.invoice_id
INNER JOIN track AS t ON i.track_id=t.track_id
INNER JOIN genre AS g ON g.genre_id = t.genre_id
GROUP BY g.name-- добавьте поле для группировки
ORDER BY tracks_purchased DESC-- добавьте поле для сортировки;
```

ЗАДАЧА 10

Отберите уникальные категории фильмов, в которых снималась Ума Вуд (англ. Uma Wood).

```
SELECT DISTINCT (c.name)
-- укажите нужные поля
FROM movie AS m
INNER JOIN film_category AS fc ON m.film_id=fc.film_id
INNER JOIN category AS c ON fc.category_id=c.category_id
INNER JOIN film_actor AS fa ON m.film_id = fa.film_id
INNER JOIN actor AS ac ON fa.actor_id = ac.actor_id
WHERE first_name = 'Uma'
```

ЗАДАЧА 11

Выведите данные обо всех треках, добавив информацию о датах, в которые эти треки покупали. Ни один трек не должен потеряться, даже если его не покупали вообще. Чтобы получить нужный результат, надо соединить три таблицы сразу, ведь таблица `invoice`, которая хранит данные о дате заказа, не содержит информации о купленных треках.

Сначала соедините таблицы `track` и `invoice_line` по ключу `track_id`, а затем присоедините таблицу `invoice` по ключу `invoice_id`. В итоговую таблицу поместите два поля: `name` из таблицы `track` и `invoice_date` из таблицы `invoice`. Приведите дату в нужный формат.

```
SELECT name,
    CAST(invoice_date AS date)   -- поместите поле с датой
FROM track AS t-- определите левую таблицу
invoice_line AS il ON t.track_id = il.track_id
LEFT OUTER JOIN invoice AS i ON il.invoice_id = i.invoice_id
```

ЗАДАЧА 12

Посчитайте для каждого года число уникальных названий купленных треков.

```
SELECT EXTRACT(YEAR FROM CAST(invoice_date AS date)) AS year_of_invoice,
      COUNT (DISTINCT(t.name))  -- посчитайте уникальное число треков
FROM track AS t
LEFT JOIN invoice_line AS il ON t.track_id = il.track_id
LEFT JOIN invoice AS i ON il.invoice_id = i.invoice_id
GROUP BY year_of_invoice-- добавьте данные для группировки;
```

ЗАДАЧА 13

Выгрузите таблицу из двух полей: первое поле с фамилией сотрудника, второе — с количеством пользователей, чьи запросы этот сотрудник обработал. Назовите поля `employee_last_name` и `all_customers` соответственно. Сгруппируйте записи по идентификатору сотрудника. Отсортируйте количество пользователей по убыванию.

```
SELECT s.last_name AS employee_last_name,
       COUNT(c.support_rep_id) AS all_customers
FROM staff AS s
LEFT OUTER JOIN client AS c ON s.employee_id = c.support_rep_id
```

```
GROUP BY s.employee_id
ORDER BY all_customers DESC
;
```

ЗАДАЧА 14

У некоторых сотрудников есть менеджеры — их идентификаторы указаны в поле `reports_to`. Посмотрите внимательно на схему базы: таблица `staff` отсылает сама к себе. Это нормально, можно не создавать новую таблицу с менеджерами.

Теперь можно разобраться в иерархии команды. Отобразите таблицу с двумя полями: в первое поле войдут фамилии всех сотрудников, а во второе — фамилии их менеджеров. Назовите поля `employee_last_name` и `manager_last_name`.

```
SELECT s.last_name AS employee_last_name,
       s1.last_name AS manager_last_name
FROM staff AS s
RIGHT OUTER JOIN staff AS s1 ON s.employee_id = s1.reports_to
;
```

ЗАДАЧА 15

Выведите идентификатор и фамилию актёра или актрисы, а также количество фильмов, в которых он или она снимались. Если нет информации, кто снимался в фильме, такой фильм тоже должен войти в таблицу.

```
SELECT ac.actor_id,
       ac.last_name,
       COUNT(m.film_id)
FROM movie AS m
LEFT OUTER JOIN film_actor AS fa ON m.film_id = fa.film_id
LEFT OUTER JOIN actor AS ac ON fa.actor_id = ac.actor_id
GROUP BY ac.actor_id
;
```

ЗАДАЧА 16

Отобразите на экране имена исполнителей, для которых в базе данных не нашлось ни одного музыкального альбома.

```
SELECT a.name
FROM artist AS a
LEFT OUTER JOIN album AS al ON a.artist_id = al.artist_id
```

```
GROUP BY a.artist_id
HAVING COUNT(al.artist_id) = 0
```

ЗАДАЧА 17

Найдите топ-40 самых длинных фильмов, аренда которых составляет больше 2 долларов. Выведите на экран название фильма (поле `title`), цену аренды (поле `rental_rate`), длительность фильма (поле `length`) и возрастной рейтинг (поле `rating`).Найдите топ-40 самых длинных фильмов, аренда которых составляет больше 2 долларов. Выведите на экран название фильма (поле `title`), цену аренды (поле `rental_rate`), длительность фильма (поле `length`) и возрастной рейтинг (поле `rating`).

```
SELECT title,
       rental_rate,
       length,
       rating
FROM movie
WHERE rental_rate > 2
ORDER BY length DESC
LIMIT 40
;
```

ЗАДАЧА 18

Проанализируйте данные о возрастных рейтингах отобранных фильмов. Выгрузите в итоговую таблицу следующие поля:

- возрастной рейтинг (поле `rating`);
- минимальное и максимальное значения длительности (поле `length`); назовите поля `min_length` и `max_length` соответственно;
- среднее значение длительности (поле `length`); назовите поле `avg_length`;
- минимум, максимум и среднее для цены просмотра (поле `rental_rate`); назовите поля `min_rental_rate`, `max_rental_rate`, `avg_rental_rate` соответственно.

Отсортируйте среднюю длительность фильма по возрастанию.

```
SELECT movie_m.rating,
       MIN(movie_m.length) AS min_length,
       MAX(movie_m.length) AS max_length,
       AVG(movie_m.length) AS avg_length,
       MIN(movie_m.rental_rate) AS min_rental_rate,
```

```
       MAX(movie_m.rental_rate) AS max_rental_rate,
       AVG(movie_m.rental_rate) AS avg_rental_rate
FROM
     (SELECT m.title,
             m.rental_rate,
           m.length,
           m.rating
      FROM movie AS m
      WHERE m.rental_rate > 2
      ORDER BY m.length DESC
      LIMIT 40
     ) AS movie_m
GROUP BY  movie_m.rating
ORDER BY avg_length
;
```

ЗАДАЧА 19

Найдите средние значения полей, в которых указаны минимальная и максимальная длительность отобранных фильмов. Отобразите только два этих поля. Назовите их `avg_min_length` и `avg_max_length` соответственно.

```
SELECT AVG(avg.min_length),
       AVG(avg.max_length)
FROM
    (SELECT top.rating,
            MIN(top.length) AS min_length,
            MAX(top.length) AS max_length,
            AVG(top.length) AS avg_length,
            MIN(top.rental_rate) AS min_rental_rate,
            MAX(top.rental_rate) AS max_rental_rate,
            AVG(top.rental_rate) AS avg_rental_rate
     FROM
         (SELECT title,
                 rental_rate,
                 length,
                 rating
          FROM movie
          WHERE rental_rate > 2
          ORDER BY length DESC
          LIMIT 40) AS top
      GROUP BY top.rating
      ORDER BY avg_length) AS avg
 ;
```

```
WITH
top AS (SELECT title,
                rental_rate,
                length,
```

```
                rating
          FROM movie
          WHERE rental_rate > 2
          ORDER BY length DESC
          LIMIT 40),

i AS (SELECT top.rating,
            MIN(top.length) AS min_length,
            MAX(top.length) AS max_length,
            AVG(top.length) AS avg_length,
            MIN(top.rental_rate) AS min_rental_rate,
            MAX(top.rental_rate) AS max_rental_rate,
            AVG(top.rental_rate) AS avg_rental_rate
            FROM top
            GROUP BY top.rating)


SELECT top.rating,
       i.min_length,
       i.max_length,
       i.avg_length,
       i.min_rental_rate,
       i.max_rental_rate,
       i.avg_rental_rate

FROM top LEFT OUTER JOIN i ON top.rating=i.rating
ORDER BY i.avg_length
```

ЗАДАЧА 20

Выведите на экран одно число — среднее количество композиций в альбомах, названия которых содержат слово `'Rock'` . В отобранных альбомах должно быть восемь треков или более.

```
SELECT AVG(rock.count_t)
FROM
     (SELECT a.title,
             COUNT(t.track_id) AS count_t
      FROM album AS a
      LEFT OUTER JOIN track AS t ON a.album_id = t.album_id
      WHERE a.title LIKE '%Rock%'
      GROUP BY a.title
      HAVING COUNT(t.track_id) >= 8) AS rock
;
```

ЗАДАЧА 21

Для каждой страны посчитайте среднюю стоимость заказов в 2009 году. Отберите данные за 2, 5, 7 и 10 месяцы и сложите средние значения стоимости заказов.

Выведите названия стран, у которых это число превышает 10 долларов.

```
SELECT m.billing_country
FROM
    (SELECT billing_country,
            AVG(total) AS avg_total
     FROM invoice
     WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = 2009
     AND EXTRACT(MONTH FROM CAST(invoice_date AS date)) IN (2, 5, 7, 10)
     GROUP BY billing_country,
              EXTRACT(MONTH FROM CAST(invoice_date AS date))
) AS m
GROUP BY billing_country
HAVING SUM(m.avg_total)  > 10
;
```

```
SELECT AVG(unit_price)
FROM invoice_line
WHERE invoice_id IN (SELECT invoice_id
                     FROM invoice_line
                     GROUP BY invoice_id)
;
```

ЗАДАЧА 22

Для каждой страны (поле `billing_country`) посчитайте минимальное, максимальное и среднее значение выручки из поля `total`. Назовите поля так: `min_total`, `max_total` и `avg_total`. Нужные поля для выгрузки хранит таблица `invoice`.

При подсчёте учитывайте только те заказы, которые включают более пяти треков. Стоимость заказа должна превышать среднюю цену одного трека.

Отсортируйте итоговую таблицу по значению в поле `avg_total` от большего к меньшему.

```
SELECT billing_country,
      MIN(total) AS min_total,
      MAX(total) AS max_total,
      AVG(total) AS avg_total
FROM invoice AS i
WHERE invoice_id IN (SELECT invoice_id
                     FROM invoice_line
                     GROUP BY invoice_id
                     HAVING COUNT(track_id) > 5)
AND total > (SELECT AVG(unit_price)
```

```
                    FROM invoice_line
                    HAVING COUNT(track_id) > 5)
GROUP BY billing_country
ORDER BY avg_total DESC
```

ЗАДАЧА 23

Отберите десять самых коротких по продолжительности треков и выгрузите названия их жанров.

```
SELECT name
FROM genre
WHERE genre_id IN (SELECT genre_id
                   FROM track
                   ORDER BY milliseconds
                   LIMIT 10)
;
```

ЗАДАЧА 24

Выгрузите уникальные названия городов, в которых стоимость заказов превышает среднее значение за 2009 год.

```
SELECT billing_city
FROM invoice
WHERE total > (SELECT AVG(total)
               FROM invoice
                WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2009')
GROUP BY billing_city
;
```

ЗАДАЧА 25

Найдите возрастной рейтинг с самыми дорогими для аренды фильмами. Для этого посчитайте среднюю стоимость аренды фильма каждого рейтинга. Выведите на экран названия категорий фильмов с этим рейтингом. Добавьте второе поле со средним значением продолжительности фильмов.

```
SELECT name,
      AVG(length) AS length
FROM movie AS m
INNER JOIN film_category AS fc ON m.film_id = fc.film_id
INNER JOIN category AS c ON fc.category_id = c.category_id
WHERE rating IN (SELECT rating
```

```
                FROM movie
                GROUP BY rating
                ORDER BY AVG(rental_rate) DESC
                LIMIT 1)

GROUP BY name
ORDER BY AVG(length) DESC
;
```

ЗАДАЧА 26

Составьте сводную таблицу. Посчитайте заказы, оформленные за каждый месяц в течение нескольких лет: с 2011 по 2013 год. Итоговая таблица должна включать четыре поля: `invoice_month`, `year_2011`, `year_2012`, `year_2013`. Поле `month` должно хранить месяц в виде числа от 1 до 12.

Если в какой-либо месяц заказы не оформляли, номер такого месяца всё равно должен попасть в таблицу.

```
SELECT y11.invoice_month,
       year_2011,
       year_2012,
       year_2013
FROM
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month,
       COUNT(invoice_id) AS year_2011
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2011'
GROUP BY invoice_month) AS y11
LEFT OUTER JOIN
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month,
       COUNT(invoice_id) AS year_2012
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2012'
GROUP BY invoice_month) AS y12 ON y11.invoice_month = y12.invoice_month
LEFT OUTER JOIN
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month,
       COUNT(invoice_id) AS year_2013
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2013'
GROUP BY invoice_month) AS y13 ON y12.invoice_month = y13.invoice_month
;
```

ЗАДАЧА 27

Отберите фамилии пользователей, которые:

- оформили хотя бы один заказ в январе 2013 года,

- а также сделали хотя бы один заказ в остальные месяцы того же года.

Пользователей, которые оформили заказы только в январе, а в остальное время ничего не заказывали, в таблицу включать не нужно.

```
SELECT last_name
FROM client AS c
LEFT OUTER JOIN invoice AS i ON c.customer_id = i.customer_id
WHERE c.customer_id IN (
                        SELECT customer_id
                        FROM invoice
                        WHERE DATE_TRUNC('month', invoice_date::timestamp) = '2013-01-01'
                        )
AND c.customer_id IN (
                              SELECT customer_id
                              FROM invoice
                              WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2013'
                              AND DATE_TRUNC('month', invoice_date::timestamp) != '2013-01-01'
                              )
GROUP BY last_name
;
```

ЗАДАЧА 28

Сформируйте статистику по категориям фильмов. Отобразите в итоговой таблице два поля:

- название категории;
- число фильмов из этой категории.

Фильмы для второго поля нужно отобрать по условию. Посчитайте фильмы только с теми актёрами и актрисами, которые больше семи раз снимались в фильмах, вышедших после 2013 года.

Назовите поля `name_category` и `total_films` соответственно. Отсортируйте таблицу по количеству фильмов от большего к меньшему, а затем по полю с названием категории в лексикографическом порядке.

```
SELECT c.name AS name_category,
       COUNT (DISTINCT m.film_id) AS total_films
FROM movie AS m
INNER JOIN film_category AS fc ON fc.film_id = m.film_id
INNER JOIN category AS c ON c.category_id = fc.category_id
INNER JOIN film_actor AS fa ON fa.film_id = fc.film_id
INNER JOIN actor AS a ON a.actor_id = fa.actor_id
WHERE a.actor_id IN (
```

```
         SELECT a.actor_id
           FROM movie AS m
              INNER JOIN film_category AS fc ON fc.film_id = m.film_id
              INNER JOIN category AS c ON c.category_id = fc.category_id
              INNER JOIN film_actor AS fa ON fa.film_id = fc.film_id
              INNER JOIN actor AS a ON a.actor_id = fa.actor_id
           WHERE  m.release_year > 2013
           GROUP BY a.actor_id
           HAVING count (m.film_id) > 7
                     )
GROUP BY name_category
ORDER BY total_films DESC, name_category asc
```

ЗАДАЧА 29

Определите, летом какого года общая выручка была максимальной. Выгрузите таблицу, в которую войдут поля:

- `country` — название страны;
- `total_invoice` — число заказов, оформленных в этой стране в тот год, когда выручка за лето была максимальной;
- `total_customer` — число клиентов, зарегистрированных в этой стране.

Отсортируйте таблицу по убыванию значений в поле `total_invoice`, а затем добавьте сортировку по названию страны в лексикографическом порядке.

```
SELECT country1.country AS country,
       country1.total_invoice AS total_invoice,
       country2.total_customer AS total_customer
FROM(
     SELECT billing_country AS country,
            COUNT(invoice_id) AS total_invoice
     FROM invoice AS i
     WHERE EXTRACT(YEAR FROM CAST(i.invoice_date AS date)) = (
                     SELECT EXTRACT(YEAR FROM CAST(invoice_date AS date))
                     FROM invoice
                     WHERE EXTRACT(MONTH FROM CAST(invoice_date AS date)) IN ('6', '7', '8')
                     GROUP BY EXTRACT(YEAR FROM CAST(invoice_date AS date))
                     ORDER BY SUM(total) DESC
                     LIMIT 1)
GROUP BY EXTRACT(YEAR FROM CAST(invoice_date AS date)),
billing_country) AS country1
LEFT OUTER JOIN(
               SELECT COUNT(customer_id) AS total_customer,
                      country
                      FROM client
                      GROUP BY country) AS country2 ON country1.country = country2.country
```

```
ORDER BY total_invoice DESC,
         country
```

ЗАДАЧА 30

Проанализируйте данные из таблицы `invoice` за 2012 и 2013 годы. В итоговую таблицу должны войти поля:

- `month` — номер месяца;
- `sum_total_2012` — выручка за этот месяц в 2012 году;
- `sum_total_2013` — выручка за этот месяц в 2013 году;
- `perc` — процент, который отображает, насколько изменилась месячная выручка в 2013 году по сравнению с 2012 годом.

Округлите значение в поле `perc` до ближайшего целого числа. Отсортируйте таблицу по значению в поле `month` от меньшего к большему.

```
SELECT month.invoice_month AS month,
      sum_total_2012.sum AS sum_total_2012,
      sum_total_2013.sum AS sum_total_2013
FROM
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) IN ('2012', '2013')
GROUP BY invoice_month) AS month

LEFT OUTER JOIN
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month,
       SUM(total) AS sum
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2012'
GROUP BY invoice_month) AS sum_total_2012 ON month.invoice_month = sum_total_2012.invoice_month

LEFT OUTER JOIN
(SELECT EXTRACT(MONTH FROM CAST(invoice_date AS date)) AS invoice_month,
       SUM(total) AS sum
FROM invoice
WHERE EXTRACT(YEAR FROM CAST(invoice_date AS date)) = '2013'
GROUP BY invoice_month) AS sum_total_2013 ON month.invoice_month = sum_total_2013.invoice_month

ORDER BY month
```

ЗАДАЧА 31

Выгрузите таблицу с десятью самыми активными инвестирующими странами. Активность страны определите по среднему количеству компаний, в которые инвестируют фонды этой страны.

Для каждой страны посчитайте минимальное, максимальное и среднее число компаний, в которые инвестировали фонды, основанные с 2010 по 2012 год включительно.

Исключите из таблицы страны с фондами, у которых минимальное число компаний, получивших инвестиции, равно нулю. Отсортируйте таблицу по среднему количеству компаний от большего к меньшему.

Для фильтрации диапазона по годам используйте оператор `BETWEEN`.

```
SELECT country_code,
       MIN(invested_companies),
       MAX(invested_companies),
       AVG(invested_companies)
FROM fund
WHERE EXTRACT(YEAR FROM CAST(founded_at AS date)) BETWEEN 2010 AND 2012
GROUP BY country_code
HAVING  MIN(invested_companies) != 0
ORDER BY AVG(invested_companies) DESC
LIMIT 10;
```

ЗАДАЧА 32

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.

```
SELECT c.name,
       COUNT(DISTINCT(e.instituition)) AS i
FROM  people AS p
INNER JOIN company AS c ON p.company_id = c.id
LEFT OUTER JOIN education AS e ON p.id = e.person_id
GROUP BY c.name
ORDER BY i DESC
LIMIT 5
;
```

ЗАДАЧА 33

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

```
SELECT DISTINCT(c.name)
FROM funding_round AS f
LEFT OUTER JOIN company AS c ON f.company_id = c.id
WHERE  c.status = 'closed' AND f.is_first_round = 1 AND f.is_last_round = 1
;
```

ЗАДАЧА 34

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

```
SELECT DISTINCT(p.id)
FROM people AS p
LEFT OUTER JOIN company AS c ON p.company_id = c.id
WHERE c.name IN (SELECT DISTINCT(c.name)
        FROM funding_round AS f
        LEFT OUTER JOIN company AS c ON f.company_id = c.id
        WHERE  c.status = 'closed' AND f.is_first_round = 1 AND f.is_last_round = 1)
 ;
```

ЗАДАЧА 35

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

```
SELECT DISTINCT (p.id),
       e.instituition
FROM people AS p
LEFT OUTER JOIN education AS e ON p.id = e.person_id
WHERE p.id IN (SELECT DISTINCT(p.id)
      FROM people AS p
      LEFT OUTER JOIN company AS c ON p.company_id = c.id
      WHERE c.name IN (SELECT DISTINCT(c.name)
        FROM funding_round AS f
        LEFT OUTER JOIN company AS c ON f.company_id = c.id
        WHERE  c.status = 'closed' AND f.is_first_round = 1 AND f.is_last_round = 1))
AND  e.instituition IS NOT NULL
;
```

ЗАДАЧА 36

Посчитайте количество учебных заведений для каждого сотрудника из предыдущего задания.

```
SELECT p.id,
       COUNT(e.instituition)
       --e.instituition
FROM people AS p
LEFT OUTER JOIN education AS e ON p.id = e.person_id
WHERE p.id IN (SELECT DISTINCT(p.id)
      FROM people AS p
      LEFT OUTER JOIN company AS c ON p.company_id = c.id
      WHERE c.name IN (SELECT DISTINCT(c.name)
        FROM funding_round AS f
        LEFT OUTER JOIN company AS c ON f.company_id = c.id
        WHERE  c.status = 'closed' AND f.is_first_round = 1 AND f.is_last_round = 1))
AND  e.instituition IS NOT NULL
GROUP BY p.id
;
```

ЗАДАЧА 37

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

```
SELECT AVG(i.instituition)
FROM(
SELECT p.id,
       COUNT(e.instituition) AS instituition
       --e.instituition
FROM people AS p
LEFT OUTER JOIN education AS e ON p.id = e.person_id
WHERE p.id IN (SELECT DISTINCT(p.id)
      FROM people AS p
      LEFT OUTER JOIN company AS c ON p.company_id = c.id
      WHERE c.name IN (SELECT DISTINCT(c.name)
        FROM funding_round AS f
        LEFT OUTER JOIN company AS c ON f.company_id = c.id
        WHERE  c.status = 'closed' AND f.is_first_round = 1 AND f.is_last_round = 1))
AND  e.instituition IS NOT NULL
GROUP BY p.id) AS i
;
```

ЗАДАЧА 38

Напишите похожий запрос: выведите среднее число учебных заведений, которые окончили сотрудники компании Facebook.

```
SELECT AVG(i.instituition)
FROM(
```

```
     SELECT p.id,
       COUNT(e.instituition) AS instituition
       FROM people AS p
       LEFT OUTER JOIN education AS e ON p.id = e.person_id
       WHERE p.id IN (
                      SELECT p.id
                      FROM people AS p
                      LEFT OUTER JOIN company AS c ON p.company_id = c.id
                      WHERE c.name  = 'Facebook')
    AND  e.instituition IS NOT NULL
    GROUP BY p.id)  AS i
;
```

ЗАДАЧА 39

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
c AS (SELECT id,
             name
      FROM company
      WHERE milestones > 6),
fr AS (SELECT id,
              company_id,
              raised_amount,
              EXTRACT(YEAR FROM CAST(funded_at AS date)) AS funded_at
       FROM funding_round
       WHERE EXTRACT(YEAR FROM CAST(funded_at AS date)) BETWEEN 2012 AND 2013),
f AS (SELECT id,
             name
      FROM fund)

SELECT f.name AS name_of_fund,
       c.name AS name_of_company,
       fr.raised_amount AS amount
FROM investment AS i
JOIN c ON  i.company_id = c.id
JOIN f ON i.fund_id = f.id
JOIN fr ON  i.funding_round_id = fr.id
WHERE c.name IS NOT NULL
AND fr.raised_amount IS NOT NULL
```

```
AND f.name IS NOT NULL
;
```

ЗАДАЧА 40

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 алфавитном порядке. Ограничьте таблицу первыми десятью записями.

```
WITH
a AS (SELECT a.id,
             acquiring_company_id,
             name,
             price_amount
      FROM acquisition AS a
      LEFT OUTER JOIN company AS c ON a.acquiring_company_id = c.id
      WHERE a.price_amount != 0),
c AS (SELECT a.id,
             acquired_company_id,
             name AS company_name,
             funding_total
      FROM acquisition AS a
      LEFT OUTER JOIN company AS c ON a.acquired_company_id = c.id
      WHERE c.funding_total != 0)

SELECT  a.name AS found_name,
        a.price_amount AS price_amount,
        c.company_name AS company_name,
        c.funding_total,
        ROUND(a.price_amount/c.funding_total) AS exit
FROM acquisition AS f
JOIN a ON f.id = a.id
JOIN c ON f.id = c.id
ORDER BY a.price_amount DESC,
```

```
         c.company_name
LIMIT 10
;
```

ЗАДАЧА 41

Выгрузите таблицу, в которую войдут названия компаний из категории `social`, получившие финансирование с 2010 по 2013 год. Выведите также номер месяца, в котором проходил раунд финансирования.

```
SELECT name,
       EXTRACT(MONTH FROM CAST(funded_at AS date)) AS month
FROM company AS c
LEFT OUTER JOIN funding_round AS fr ON c.id = fr.company_id
WHERE category_code = 'social'
AND EXTRACT(YEAR FROM CAST(funded_at AS date)) BETWEEN 2010 AND 2013
;
```

ЗАДАЧА 42

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
i1 AS (SELECT EXTRACT(MONTH FROM CAST(acquired_at AS date)) AS month,
       COUNT(acquired_company_id) AS company_count,
       SUM(price_amount) AS total
FROM acquisition
WHERE EXTRACT(YEAR FROM CAST(acquired_at AS date)) BETWEEN 2010 AND 2013
GROUP BY EXTRACT(MONTH  FROM CAST(acquired_at AS date))),

i2 AS (SELECT EXTRACT(MONTH FROM CAST(funded_at AS date)) AS month,
       COUNT(DISTINCT(f.name)) AS found_count
FROM fund AS f
LEFT JOIN investment AS i ON f.id  = i.fund_id
LEFT JOIN funding_round AS fr ON i.funding_round_id  = fr.id
WHERE EXTRACT(YEAR FROM CAST(funded_at AS date)) BETWEEN 2010 AND 2013
```

```
AND f.country_code = 'USA'
GROUP BY EXTRACT(MONTH  FROM CAST(funded_at AS date)))

SELECT i1.month,
       i2.found_count,
       i1.company_count,
       i1.total
FROM i1
JOIN i2 ON i1.month = i2.month
;
```

ЗАДАЧА 43

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.

```
SELECT Y11.country_code,
       AVG2011,
       AVG2012,
       AVG2013
FROM(
SELECT country_code,
       AVG(funding_total) AS AVG2011
FROM company
WHERE EXTRACT(YEAR FROM CAST(founded_at AS date)) = 2011
GROUP BY country_code) AS Y11
JOIN
(SELECT country_code,
       AVG(funding_total) AS AVG2012
FROM company
WHERE EXTRACT(YEAR FROM CAST(founded_at AS date)) = 2012
GROUP BY country_code) AS Y12 ON Y11.country_code = Y12.country_code
JOIN
(SELECT country_code,
       AVG(funding_total) AS AVG2013
FROM company
WHERE EXTRACT(YEAR FROM CAST(founded_at AS date)) = 2013
GROUP BY country_code) AS Y13 ON Y12.country_code = Y13.country_code
ORDER BY AVG2011 DESC
;
```

Список дополнительных материалов

- Руководство по стилю SQL — основные рекомендации по стилю и оформлению запросов.

- Памятка/шпаргалка по SQL — объёмная структурированная статья, в которой собрали основные понятия, операторы и методы работы с базой данных.
- Нормализация реляционных баз данных — статья о том, как, пользуясь важными понятиями реляционной теории, можно спроектировать логичную базу данных.
- Нормализация отношений. Шесть нормальных форм — материал о том, какие методы нормализации использовать при проектировании базы данных.