**Курсовая работа “Моделирование изменения балансов студентов”**

Наш курс по SQL подошел к концу. Чтобы структурировать полученные знания, проверить себя и собрать небольшой кейс в портфолио, предлагаем выполнить курсовую работу.

**Для чего это всё?**

**Чтобы проверить себя по следующим пунктам:**

* [ ] Умение писать запросы без логических и синтаксических ошибок.
* [ ] Умение видеть в данных проблемы и говорить о них.
* [ ] Умение писать код, который выводит правильный результат в том виде, в котором его запросил заказчик.
* [ ] Умение писать читаемый код, который может легко понять коллега, руководитель или ревьюер.

**Отличным бонусом будет, если вы:**

* [ ] сделаете визуализации и на ее основании соберете гипотезы.
* **В каком формате сдавать курсовую?**

Соберите единый файл Excel, в котором будут данные о балансах, добавьте туда же ссылку на ваш запрос из Metabase (как вы делали это в домашних работах).

**На отдельных листах:**

* + соберите вопросы к дата-инженерам и владельцам таблицы payments (назовите этот лист **Задание 1**);
  + соберите выводы из визуализации (линейной диаграммы) получившегося результата (назовите этот лист **Задание 2**).

**Основное задание курсовой**

**Наша задача — смоделировать изменение балансов студентов.**

Баланс — это количество уроков, которое есть у каждого студента.

Чтобы проверить, всё ли в порядке с нашими данными, составьте список гипотез и вопросов.

Нам важно понимать:

* сколько всего уроков было на балансе **всех учеников** за каждый календарный день;
* как это количество менялось под влиянием транзакций (оплат, начислений, корректирующих списаний) и уроков (списаний с баланса по мере прохождения уроков).

Также мы хотим создать таблицу, где будут балансы **каждого студента** за каждый день.

**Результат курсовой**

В результате должен получиться запрос, который собирает данные о балансах студентов за каждый "прожитый" ими день.

**План решения курсового проекта**

Это пошаговый план, который поможет вам не запутаться в последовательности написания запросов и подзапросов.

**Шаг 1**

Узнаем, когда была первая транзакция для каждого студента. Начиная с этой даты, мы будем собирать его баланс уроков. **Создадим CTE** first\_payments с двумя полями: user\_id и first\_payment\_date (дата первой успешной транзакции).

**Шаг 2**

Соберем таблицу с датами за каждый календарный день 2016 года. Есть разные способы это сделать, но мы воспользуемся тем, который уже знаем. Выберем все даты из таблицы classes, **создадим CTE** all\_dates с полем dt, где будут храниться уникальные даты (без времени) уроков.

**Шаг 3**

Узнаем, за какие даты имеет смысл собирать баланс для каждого студента. Для этого объединим таблицы и создадим CTE all\_dates\_by\_user, где будут храниться все даты жизни студента после того, как произошла его первая транзакция. **В таблице** должны быть такие поля: user\_id, dt.

Для выполнения этого шага в таком виде вам может пригодится следующая теория:

[JOIN по неравенству](https://www.notion.so/JOIN-ff9e5160928042268679a7d4b6541c7e?pvs=21)

**Шаг 4**

Найдем все изменения балансов, связанные с успешными транзакциями. Выберем все транзакции из таблицы payments, сгруппируем их по user\_id и дате транзакции (без времени) и найдем сумму по полю classes. **В результате** получим CTE payments\_by\_dates с полями: user\_id, payment\_date, transaction\_balance\_change (сколько уроков было начислено или списано в этот день).

**Шаг 5**

Найдем баланс студентов, который сформирован только транзакциями. Для этого объединим all\_dates\_by\_user и payments\_by\_dates так, чтобы совпадали даты и user\_id. Используем оконные выражения (функцию sum), чтобы найти кумулятивную сумму по полю transaction\_balance\_change для всех строк до текущей включительно с разбивкой по user\_id и сортировкой по dt. **В результате** получим CTE payments\_by\_dates\_cumsum с полями: user\_id, dt, transaction\_balance\_change — transaction\_balance\_change\_cs (кумулятивная сумма по transaction\_balance\_change). При подсчете кумулятивной суммы можно заменить пустые значения нулями.

**Шаг 6**

Найдем изменения балансов из-за прохождения уроков. Создадим CTE classes\_by\_dates, посчитав в таблице classes количество уроков за каждый день для каждого ученика. Нас не интересуют вводные уроки и уроки со статусом, отличным от success и failed\_by\_student. **Получим результат** с такими полями: user\_id, class\_date, classes (количество пройденных в этот день уроков). Причем classes мы умножим на -1, чтобы отразить, что - — это списания с баланса.

**Шаг 7**

По аналогии с уже проделанным шагом для оплат создадим CTE для хранения кумулятивной суммы количества пройденных уроков. Для этого объединим таблицы all\_dates\_by\_user и classes\_by\_dates так, чтобы совпадали даты и user\_id. Используем оконные выражения (функцию sum), чтобы найти кумулятивную сумму по полю classes для всех строк до текущей включительно с разбивкой по user\_id и сортировкой по dt. **В результате** получим CTE classes\_by\_dates\_dates\_cumsumс полями: user\_id, dt, classes — classes\_cs(кумулятивная сумма по classes). При подсчете кумулятивной суммы обязательно нужно заменить пустые значения нулями.

**Шаг 8**

Создадим CTE balances \*\*\*\*с вычисленными балансами каждого студента. Для этого объединим таблицы payments\_by\_dates\_cumsum \*\*\*\*и classes\_by\_dates\_dates\_cumsum так, чтобы совпадали даты и user\_id.

**Получим такие поля:** user\_id, dt, transaction\_balance\_change, transaction\_balance\_change\_cs, classes, classes\_cs, balance (classes\_cs + transaction\_balance\_change\_cs).

**Задание 1**

Выберите топ-1000 строк из CTE balances с сортировкой по user\_id и dt. Посмотрите на изменения балансов студентов.

Какие вопросы стоит задавать дата-инженерам и владельцам таблицы payments?

**Шаг 9**

Посмотрим, как менялось общее количество уроков на балансах студентов.

Для этого просуммируем поля transaction\_balance\_change, transaction\_balance\_change\_cs, classes, classes\_cs, balance из CTE balances с группировкой и сортировкой по dt.

**Задание 2**

Создайте визуализацию (линейную диаграмму) итогового результата.

Какие выводы можно сделать из получившейся визуализации?

## Полезная теория

Перед курсовой рекомендуем почитать короткую шпагалку про то, как использовать JOIN по неравенству. Это то, что пригодится при решении задачи по указанному плану:

[JOIN по неравенству](https://www.notion.so/JOIN-ff9e5160928042268679a7d4b6541c7e?pvs=21)

Про все остальные виды JOIN-ов мы говорили в четвертом и пятом уроках:

[4.1. Функции JOIN: учимся объединять таблицы правильно](https://www.notion.so/4-1-JOIN-cde250b0677c4dbd9ca12a2a59a14d4a?pvs=21)

[5.1. Функции JOIN и кое-что еще](https://www.notion.so/5-1-JOIN-88865210f2ad4f5f9f04363027c2c7dc?pvs=21)

Также можно вспомнить, что такое подзапросы и почему они так важны:

[6.1. Подзапросы и WITH](https://www.notion.so/6-1-WITH-8746533f74254032986666de48e3a961?pvs=21)