## Задания на SQL

Решал одним вечером, поэтому не проверял решения, это больше концептуальное и надо проверять, тюнить на реальных данных

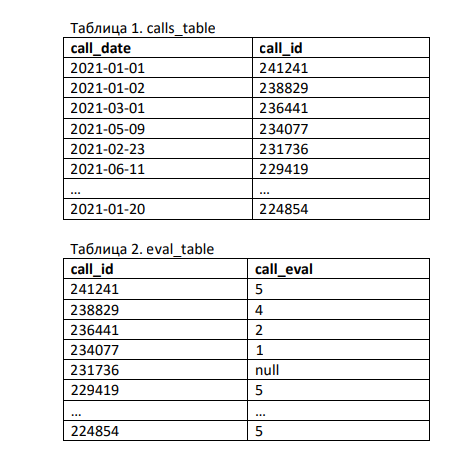
**Задача №1**

Даны две таблицы. Первая таблица calls\_table содержит данные о дате звонка. Вторая таблица eval\_table содержит данные с оценкой данного звонка.

Нужно написать SQL-запрос, выведет для каждого дня долю (%) плохих оценок от всех проставленных оценок в этот день.

Звонок оценивается баллами от 1 до 5, плохой оценкой считается оценка от 3 и ниже.

По некоторым звонкам оценка может быть не проставлена, в таком случае в таблице для такого звонка будет значение null в поле с оценкой.



Решение

Можно пойти двумя путями

Первый путь

Объединить таблицы по call\_id.

Отфильтровать звонки с оценками.

Определить плохие оценки (от 1 до 3).

Рассчитать долю плохих оценок от всех оценок за каждый день.

SELECT c.call\_date,

ROUND(SUM(CASE WHEN e.call\_eval <= 3 THEN 1 ELSE 0 END) \* 100.0 / COUNT(e.call\_eval), 2) AS bad\_eval\_percentage

FROM calls\_table c

LEFT JOIN eval\_table e ON c.call\_id = e.call\_id

WHERE e.call\_eval IS NOT NULL -- исключаем звонки без оценок

GROUP BY c.call\_date

ORDER BY c.call\_date;

Второй путь

1. Добавить в таблицу с оценками колонку «плохие»
2. Затем уже эту таблиц приджоинить

--CTE-запрос

With subquery as (

SELECT \* FROM eval\_table e

CASE

WHEN e.call\_eval <= 3 THEN 1

ELSE 0

END AS bad

WHERE e.call\_eval IS NOT NULL -- исключаем звонки без оценок (аналогично первому варианту)

--основной запрос

SELECT c.call\_date, count(call\_eval), count(bad), bad/count(call\_eval) as proportion

FROM call\_table c

LEFT JOIN subquery e ON c.call\_id = e.call\_id

GROUP BY c.call\_date

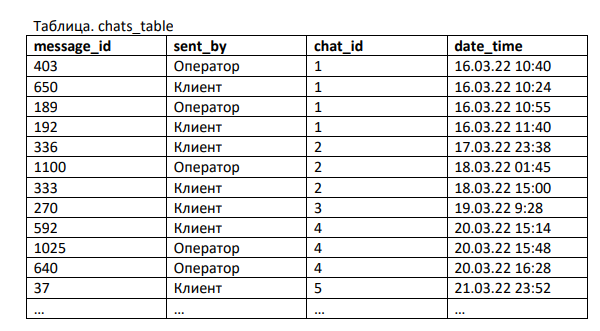
ORDER BY c.call\_date;

(в этом запросе я не прописывал округление, но можно сделать аналогично)

**Задача №2**

Таблица содержит данные с логом сообщений в чате клиента с оператором. Все чаты инициируются клиентом - первое сообщение всегда клиента, все последующие это хаотичный порядок сообщений клиента/оператора.

Нужно написать SQL-запрос, который для каждого чата вывести скорость ответа оператора т.е. через сколько секунд Клиент после своего первого сообщения получил первый ответ оператора. В случае, если ответа оператора не последовало проставить «Оператор не ответил»



Тут в целом логика такая

* Найти первое сообщение клиента для каждого чата.
* Найти первое сообщение оператора после первого сообщения клиента.
* Рассчитать разницу во времени между первым сообщением клиента и первым ответом оператора.
* Если оператор не ответил, вывести «Оператор не ответил».

WITH first\_client\_message AS (

-- Находим первое сообщение клиента в каждом чате

SELECT

chat\_id,

MIN(date\_time) AS first\_client\_time

FROM

chats\_table

WHERE

sent\_by = 'Клиент'

GROUP BY

chat\_id

),

first\_operator\_response AS (

-- Находим первое сообщение оператора после первого сообщения клиента

SELECT

c.chat\_id,

MIN(c.date\_time) AS first\_operator\_time

FROM

chats\_table c

JOIN

first\_client\_message fcm ON c.chat\_id = fcm.chat\_id

WHERE

c.sent\_by = 'Оператор'

AND c.date\_time > fcm.first\_client\_time

GROUP BY

c.chat\_id

)

-- Основной запрос

SELECT

fcm.chat\_id,

CASE

-- Если оператор ответил, выводим разницу во времени в секундах

WHEN forr.first\_operator\_time IS NOT NULL THEN TIMESTAMPDIFF(SECOND, fcm.first\_client\_time, forr.first\_operator\_time)

-- Если оператор не ответил, выводим "Оператор не ответил"

ELSE 'Оператор не ответил'

END AS response\_time

FROM

first\_client\_message fcm

LEFT JOIN

first\_operator\_response forr ON fcm.chat\_id = forr.chat\_id

ORDER BY

fcm.chat\_id;

Другой вариант немного

SELECT

cl.chat\_id,

cl.time AS client\_time,

op.time AS operator\_time,

TIMESTAMPDIFF(SECOND, cl.time, op.time) AS time\_difference -- Разница во времени в секундах

FROM (

SELECT

chat\_id,

MIN(date\_time) AS time

FROM chats\_table

WHERE sent\_by = 'Клиент'

GROUP BY chat\_id

) AS cl

LEFT JOIN (

SELECT

chat\_id,

MIN(date\_time) AS time

FROM chats\_table

WHERE sent\_by = 'Оператор'

GROUP BY chat\_id

) AS op ON cl.chat\_id = op.chat\_id;

Задача №3

Таблица call\_status\_log содержит данные лога изменения статусов звонка. Есть 4 вида статусов:

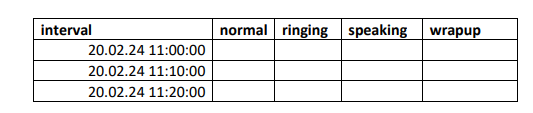
• Normal – режим ожидание звонка

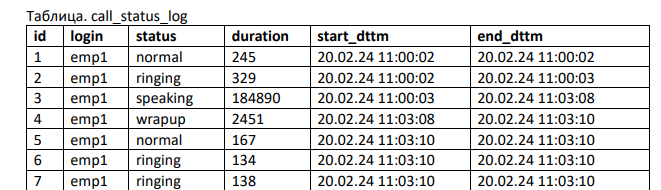
• Ringing – звон телефона до снятия трубки

• Speaking – диалог

• Wrapup – время поствызовной обработки

Ниже формат результирующей таблицы, которую нужно получить. Это интервальный отчет с разбивкой на 10 минутные интервалы и нужно показать, сколько точно времени в секундах было каждого статуса в каждом 10-ти минутном интервале.





Тут я бы решал так, как бы решал в экселе

А именно:

Добавил бы колонку с интервалом (формулой если), а потом просто сделать сводную, где статусы были бы колонками

А в SQL для этого используется CASE

Поэтому примерно такой запрос

SELECT

CASE

WHEN start\_ddtm < '2024-02-20 11:10:00' THEN '2024-02-20 11:00:00'

WHEN start\_ddtm < '2024-02-20 11:20:00' THEN '2024-02-20 11:10:00'

WHEN start\_ddtm < '2024-02-20 11:30:00' THEN '2024-02-20 11:20:00'

ELSE 'выход за пределы диапазона'

END AS interval,

SUM(CASE WHEN status = 'Normal' THEN duration ELSE 0 END) AS Normal,

SUM(CASE WHEN status = 'Ringing' THEN duration ELSE 0 END) AS Ringing,

SUM(CASE WHEN status = 'Speaking' THEN duration ELSE 0 END) AS Speaking,

SUM(CASE WHEN status = 'Wrapup' THEN duration ELSE 0 END) AS Wrapup

FROM

call\_status\_log

GROUP BY

interval

ORDER BY

interval;

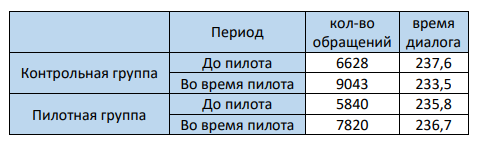
(тут простой код, он не учитывает ситуацию, когда звонок начался в одно время, а закончился в другое)

## Задания на Аналитику

**Задача №4**

В контактном центре провели пилот по изменению процесса консультации клиента с целью снижения времени диалога. Всех операторов входящего канала поделили на пилотную и контрольную группы.

В таблице приведены цифровые результаты пилота. Необходимо подготовить аналитическую справку - выводы по итогам пилота.



из предоставленной таблицы можно сделать следующие выводы:

1. **Сравнение количества обращений:**
   * В контрольной группе количество обращений увеличилось на **36,4%** (с 6628 до 9043) в период пилота.
   * В пилотной группе количество обращений также увеличилось на **33,9%** (с 5840 до 7820) в период пилота.
2. **Сравнение времени диалога:**
   * В контрольной группе время диалога **снизилось** незначительно — с **237,6** секунд до **233,5** секунд (снижение на 1,7%).
   * В пилотной группе время диалога **незначительно увеличилось** — с **235,8** секунд до **236,7** секунд (увеличение на 0,38%).
3. **Вывод по эффективности пилота:**
   * Можно предположить, что основная цель пилота - это снижение времени диалога, которая не была достигнута. В контрольной группе время диалога даже немного уменьшилось, в то время как в пилотной группе оно незначительно увеличилось.
   * Увеличение количества обращений в обеих группах могло негативно повлиять на оценку эксперимента. Но видимо объем работы вырос
4. **Рекомендации:**

Пересмотреть, передизайнить пилот, чтобы повысить его эффективность

Постараться сделать более схожими по размеру объемы до пилота и после пилота, для более корректного сравнения

(в целом, если считать это аб-тестом, то желательно подходить к этому более серьезно, учитывая все требования к подобным экспериментам)