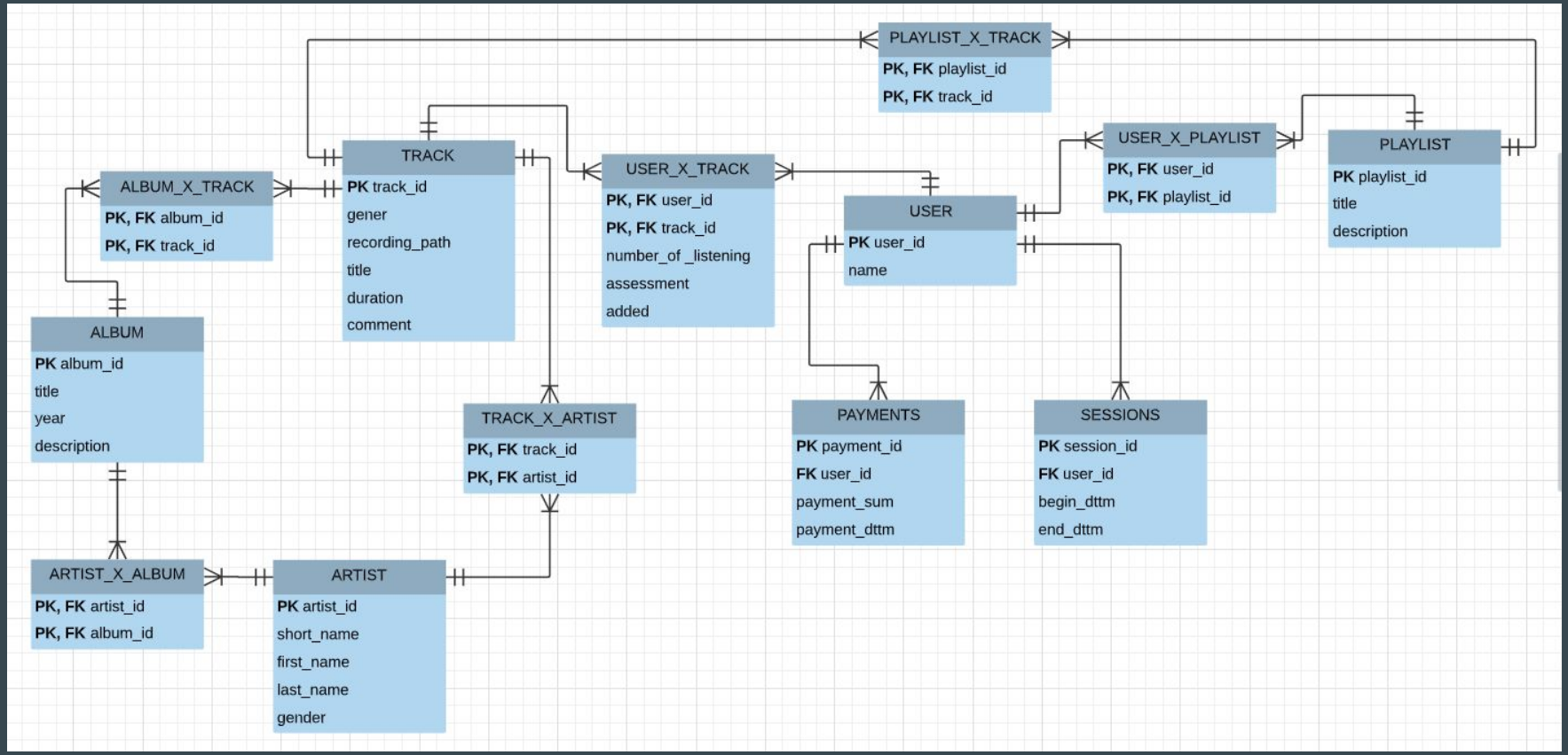
Тема проекта: Музыкальный стриминговый сервис

...

План

- Описание проекта
- DAU, PPU, Revenue
- Масштабирование

Схема данных



Некоторые запросы к БД

Любимый жанр пользователя

```
SELECT A.USER_ID, GENRE
FROM (SELECT USER_ID, GENRE, COUNT(GENRE) AS COUNT_LISTEN
      FROM Melnikova.USER_X_TRACK
               LEFT JOIN Melnikova.TRACK
                         ON Melnikova.USER_X_TRACK.TRACK_ID = Melnikova.TRACK.TRACK_ID
      GROUP BY USER_ID, GENRE) AS A
         LEFT JOIN (SELECT USER_ID, MAX(COUNT_LISTEN) AS MAX_COUNT
                    FROM (SELECT USER_ID, GENRE, COUNT(GENRE) AS COUNT_LISTEN
                          FROM Melnikova.USER_X_TRACK
                                   LEFT JOIN Melnikova.TRACK
                                             ON Melnikova.USER_X_TRACK.TRACK_ID = Melnikova.TRACK.TRACK_ID
                          GROUP BY USER_ID, GENRE) AS C
                    GROUP BY USER_ID) AS B
                   ON A.USER_ID = B.USER_ID
WHERE A.COUNT_LISTEN = B.MAX_COUNT
ORDER BY USER_ID;
```

USER_ID	GENRE
0	Pop
0	Rock
1	Jazz
1	Pop
2	Pop
3	Jazz
3	Pop
4	Rock
5	Pop
6	Pop
7	Afrikanische M…
7	Pop
7	Alternative
7	Jazz
8	Rock
9	Jazz
10	Pop
10	Rock
11	Pop
12	Electro
13	Heavy Metal
13	Jazz
13	Filme/Videospi…
13	Pop

Количество треков, выпущенных артистом, в год

```
WITH
    COUNT_TRACKS AS (
        SELECT ALBUM_ID, COUNT(TRACK_ID) AS COUNT_TRACK
        FROM Melnikova.ALBUM_X_TRACK
        GROUP BY ALBUM_ID
    ),
    ALBUM_ARTIST AS (
        SELECT ARTIST_ID, ALBUM.ALBUM_ID, YEAR(YEAR) AS YEAR
        FROM Melnikova.ALBUM
        LEFT JOIN Melnikova.ARTIST_X_ALBUM
        ON ARTIST_X_ALBUM.ALBUM_ID = ALBUM.ALBUM_ID
    )
SELECT ALBUM_ARTIST.ARTIST_ID, YEAR, SUM(COUNT_TRACK) OVER(PARTITION BY ARTIST_ID ORDER BY YEAR) AS NUM_TRACKS
FROM COUNT_TRACKS
        LEFT JOIN ALBUM_ARTIST ON ALBUM_ARTIST.ALBUM_ID = COUNT_TRACKS.ALBUM_ID
ORDER BY ARTIST_ID;
```

	ARTIST_ID	YEAR	NUM_TRACKS
1	317	2008	10
2	381	2008	2
3	409	2008	5
4	475	2008	24
5	564	2007	2
6	896	2007	2
7	1026	2008	16
8	1055	2009	13
9	1060	2000	10
10	1083	2008	6
11	1302	2005	31
12	1439	2007	14
13	1802	2002	2
14	2059	1993	12
15	2802	2008	2
16	3197	2008	3
17	4029	1996	12
18	4093	2007	15
19	5080	2008	22
20	5080	2008	22
21	5369	1999	4
22	6362	2008	1
23	7525	2008	3

DAU, PPU, REVENUE

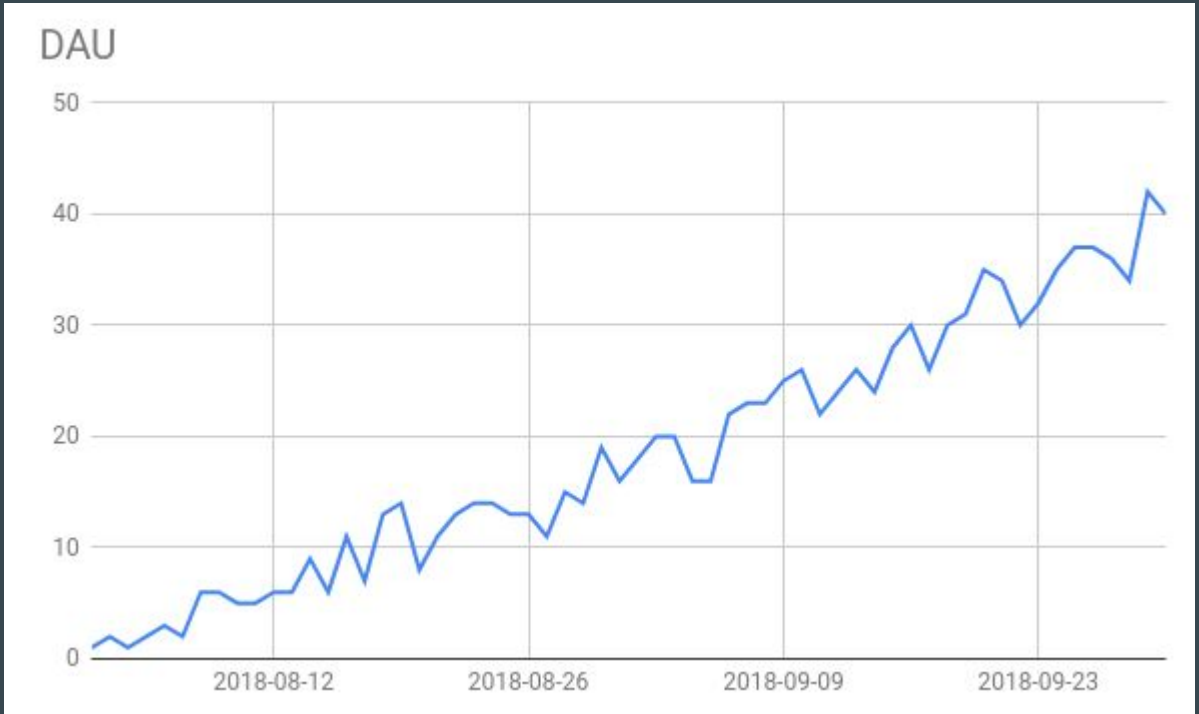
DAU запрос

```
WITH RECURSIVE CTE (DT) AS
                   (
                       SELECT MIN(CAST(BEGIN_DTTM AS DATE)) AS DT
                       FROM Melnikova.SESSIONS
                       UNION ALL
                       SELECT DT + INTERVAL 1 DAY
                       FROM CTE
                       WHERE DT + INTERVAL 1 DAY <= (SELECT MAX(CAST(BEGIN_DTTM AS DATE))
                                                     FROM (SELECT BEGIN_DTTM FROM Melnikova.SESSIONS) AS SESSIONS_DTTM_)
                   )
SELECT CTE.DT, COUNT(DISTINCT USER_ID)
FROM CTE
         LEFT JOIN (SELECT BEGIN_DTTM, USER_ID FROM Melnikova.SESSIONS) AS SESSIONS_DTTM_USER
                   ON CTE.DT = CAST(SESSIONS_DTTM_USER.BEGIN_DTTM AS DATE)
GROUP BY CTE.DT
ORDER BY CTE.DT;
```

DAU EXPLAIN

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	PRIMARY	<derived2>	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	3	100	Using filesort
2	1	PRIMARY	SESSIONS	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	3572	100	Using where
3	2	DERIVED	SESSIONS	*<null>*	index	*<null>*	dttm_index	5	*<null>*	3572	100	Using index
4	3	UNION	CTE	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	2	100	Recursive; Using where
5	5	SUBQUERY	SESSIONS	*<null>*	index	*<null>*	dttm_index	5	*<null>*	3572	100	Using index

DAU график



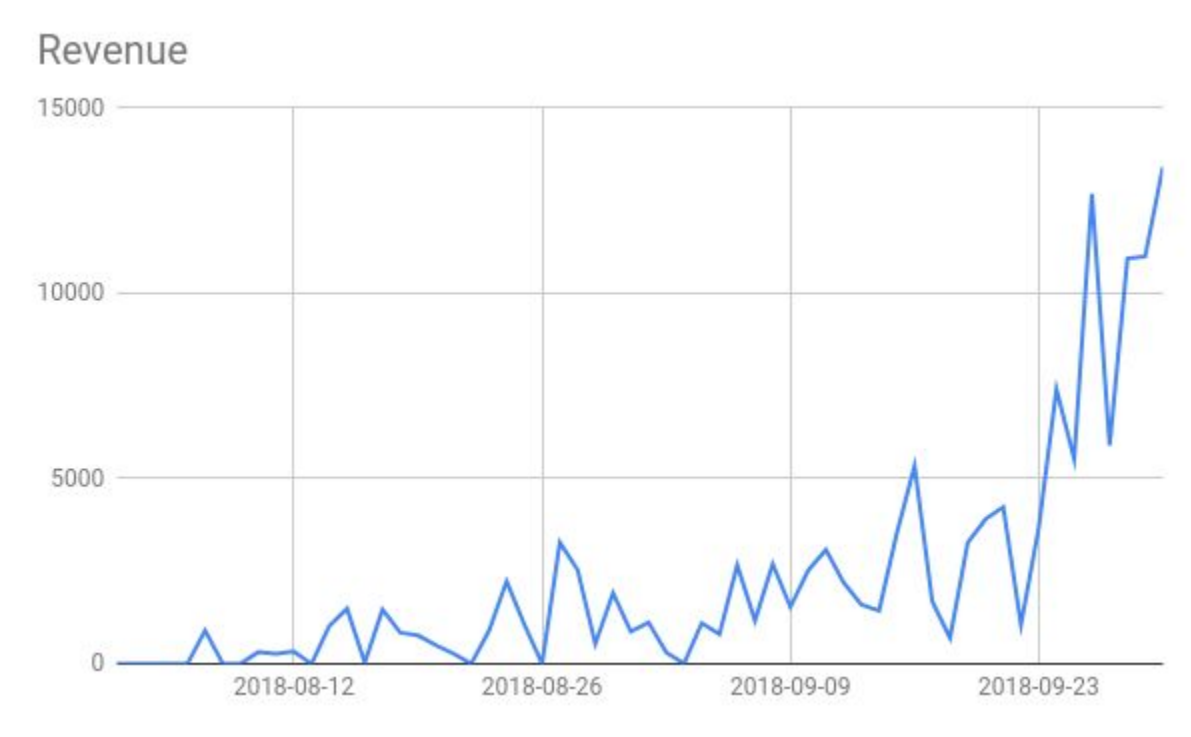
Revenue запрос

```
WITH RECURSIVE
    CTE(DT) AS (
        SELECT MIN(CAST(BEGIN_DTTM AS DATE)) AS DT
        FROM Melnikova.SESSIONS
        UNION ALL
        SELECT DT + INTERVAL 1 DAY
        FROM CTE
        WHERE DT + INTERVAL 1 DAY <= (SELECT MAX(CAST(BEGIN_DTTM AS DATE)) FROM Melnikova.SESSIONS)
    ),
    SUMM AS (
        SELECT CAST(Melnikova.PAYMENTS.PAYMENT_DTTM AS DATE) AS DT, SUM(PAYMENT_SUM) AS SUM
        FROM Melnikova.PAYMENTS
        GROUP BY DT
    )
SELECT CTE.DT, COALESCE(SUM, 0) AS SUM_PAYMENT
FROM CTE
         LEFT JOIN SUMM ON CTE.DT = SUMM.DT;
```

Revenue EXPLAIN

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	PRIMARY	<derived2>	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	3	100	*<null>*
2	1	PRIMARY	<derived6>	*<null>*	ref	<auto_key0>	<auto_key0>	4	CTE.DT	10	100	*<null>*
3	6	DERIVED	PAYMENTS	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	269	100	Using temporary
4	2	DERIVED	SESSIONS	*<null>*	index	*<null>*	dttm_index	5	*<null>*	3572	100	Using index
5	3	UNION	CTE	*<null>*	ALL	*<null>*	*<null>*	*<null>*	*<null>*	2	100	Recursive; Using where
6	5	SUBQUERY	SESSIONS	*<null>*	index	*<null>*	dttm_index	5	*<null>*	3572	100	Using index

Revenue график



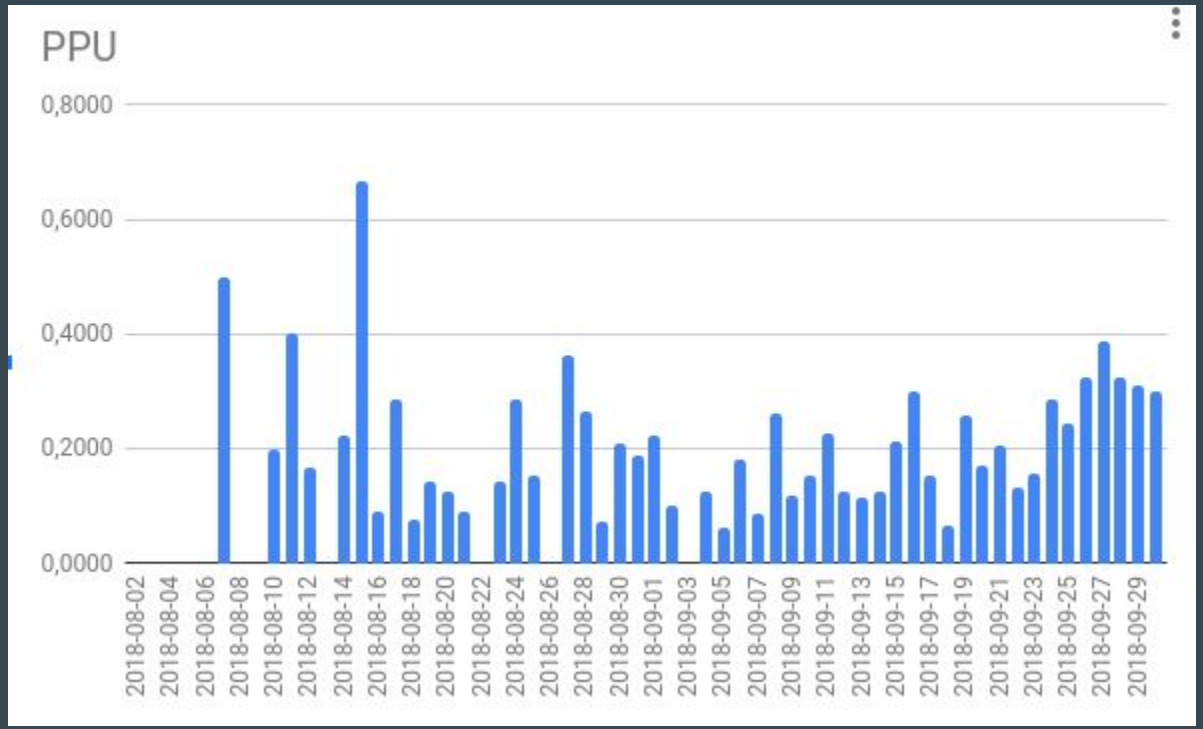
PPU запрос

```
EXPLAIN
WITH RECURSIVE
    CTE(DT) AS (
        SELECT MIN(CAST(BEGIN_DTTM AS DATE)) AS DT
        FROM Melnikova.SESSIONS
        UNION ALL
        SELECT DT + INTERVAL 1 DAY
        FROM CTE
        WHERE DT + INTERVAL 1 DAY <= (SELECT MAX(CAST(BEGIN_DTTM AS DATE)) FROM Melnikova.SESSIONS)
    ),
    DAU AS (
        SELECT CAST(Melnikova.SESSIONS.BEGIN_DTTM AS DATE) AS DT, COUNT(DISTINCT USER_ID) AS COUNT_USER
        FROM Melnikova.SESSIONS
        GROUP BY DT
    ),
    PU AS (
        SELECT CAST(Melnikova.PAYMENTS.PAYMENT_DTTM AS DATE) AS DT, COUNT(DISTINCT USER_ID) AS COUNT_PAY_USER
        FROM Melnikova.PAYMENTS
        GROUP BY DT
    )
SELECT CTE.DT, COALESCE(COUNT_PAY_USER, 0) / COALESCE(COUNT_USER, 1) AS PPU
FROM CTE
         LEFT JOIN DAU ON CTE.DT = DAU.DT
         LEFT JOIN PU ON CTE.DT = PU.DT;
```

PPU EXPLAIN

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	PRIMARY	<derived2>	<null>	ALL	<null>	<null>	<null>	<null>	3	100	<null>
2	1	PRIMARY	<derived6>	<null>	ref	<auto_key0>	<auto_key0>	4	CTE.DT	35	100	<null>
3	1	PRIMARY	<derived7>	<null>	ref	<auto_key0>	<auto_key0>	4	CTE.DT	10	100	<null>
4	7	DERIVED	PAYMENTS	<null>	ALL	<null>	<null>	<null>	<null>	269	100	Using filesort
5	6	DERIVED	SESSIONS	<null>	ALL	dttm_index	<null>	<null>	<null>	3572	100	Using filesort
6	2	DERIVED	SESSIONS	<null>	index	<null>	dttm_index	5	<null>	3572	100	Using index
7	3	UNION	CTE	<null>	ALL	<null>	<null>	<null>	<null>	2	100	Recursive; Using where
8	5	SUBQUERY	SESSIONS	<null>	index	<null>	dttm_index	5	<null>	3572	100	Using index

PPU график


PPU
0,8000
0,6000
0,4000
0,2000
0,0000
2018-08-02
2018-08-04
2018-08-06
2018-08-08
2018-08-10
2018-08-12
2018-08-14
2018-08-16
2018-08-18
2018-08-20
2018-08-22
2018-08-24
2018-08-26
2018-08-28
2018-08-30
2018-09-01
2018-09-03
2018-09-05
2018-09-07
2018-09-09
2018-09-11
2018-09-13
2018-09-15
2018-09-17
2018-09-19
2018-09-21
2018-09-23
2018-09-25
2018-09-27
2018-09-29

МАСШТАБИРОВАНИЕ

Проблемы

- Рост числа пользователей

- Решение: Разделим таблицу Users на примерно равные части и вынесем каждую часть на отдельный сервер.

  (горизонтальный шардинг)

Проблемы

- Рост числа сессий и платежей

- Решение: вынесем таблицы Sessions и Payments на отдельный сервер

  (вертикальный шардинг)

- Можно было бы хранить эти таблицы на отдельных серверах, но тогда теряем возможность быстро делать SQL запросы объединяющие эти таблицы.
- Не храним информацию о давних сессиях

  (обратное масштабирование)

Проблемы

- Высокая нагрузка на сервер из-за частых запросов к таблицам Track, Artist, Album, Playlist

- Решение: Храним копии таблиц на нескольких серверах

  (репликация)

♡ Спасибо за внимание ♡