SQL PROJECT

By Yuvraj Giri

Introduction

This is the project of Song website Data analysis. The database in this project contains two tables i.e events and songs.

Introducing myself

My name is Yuvraj Giri. Currently I'm learning SQL and have upper intermediate skills. This is the project of song website data analysis and also my Fifth Project. I've included various questions on this project which will be beneficial for making data driven business decisions and extracting meaningful insights from this.

QUESTIONS

Find the most playes song in a session

It helps in understanding user engagement and user preference patterns, which can be used for targeted advertising.

Count how many distinct users listened to a specific artist

It helps in Artist Popularity Measurement, which will be useful for making targeted Artist Campaigns and helps in Revenue Generation.

Identify the top 5 most active users (by number of sessions)

We can do targeted marketing, promotions and personalized content recommendations by utilizing insights gained from understanding power users.

Calculate the average session length for users who listen to premium (paid) content

Understanding premium user engagement can support business decision-making, including revenue growth and cost-benefit analysis.

QUESTIONS

Top 5 location by number of music listen

It helps the platform understand regional preferences, enabling targeted marketing, enhancing content strategy, optimizing user experience and informing business decisions.

Find artists whose songs are skipped the most

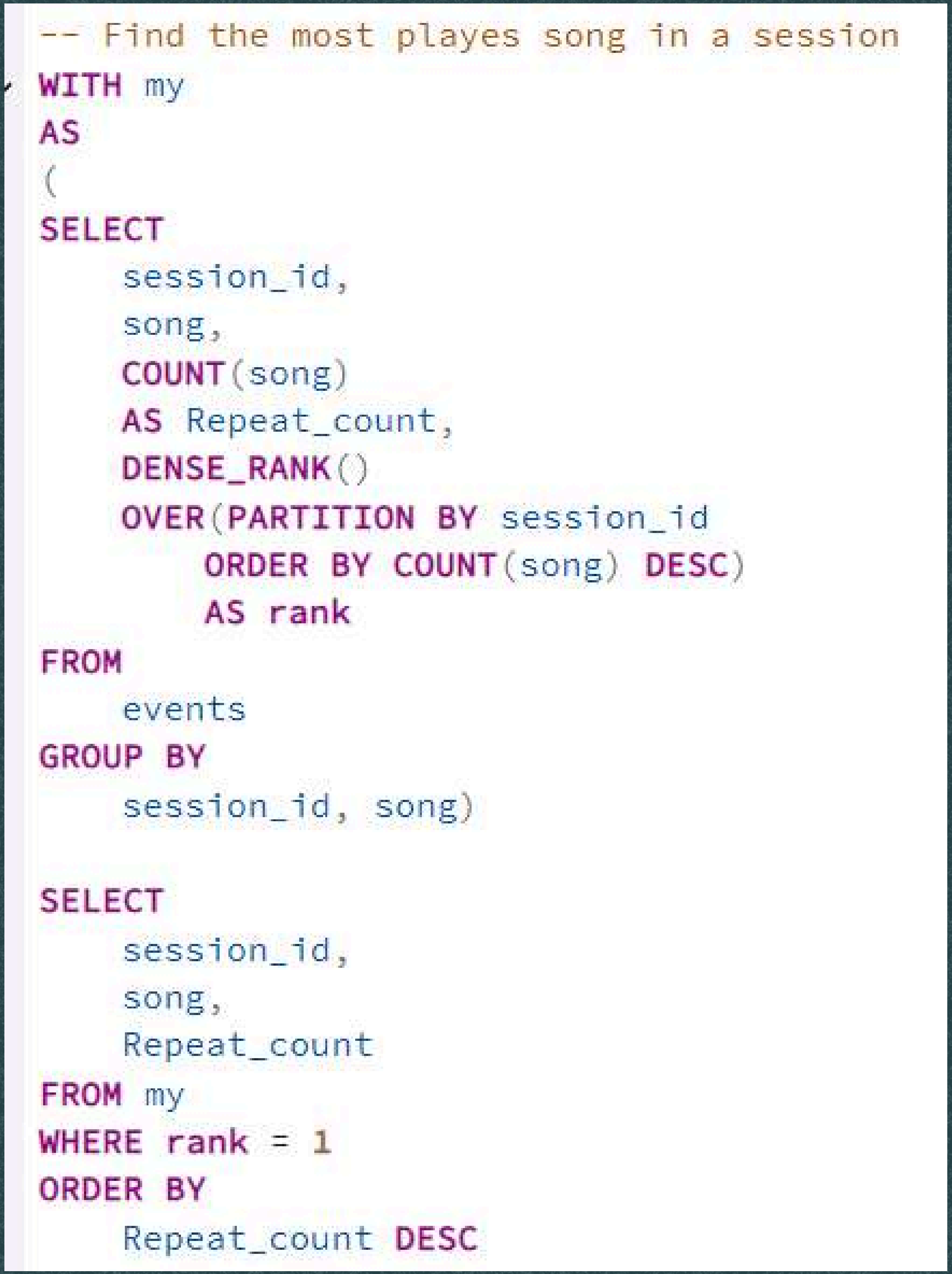
It helps in assessing content quality, improving user experience, providing feedback to artists, optimizing recommendation algorithms, and guiding advertising and revenue strategies.

Count of songs by Time period

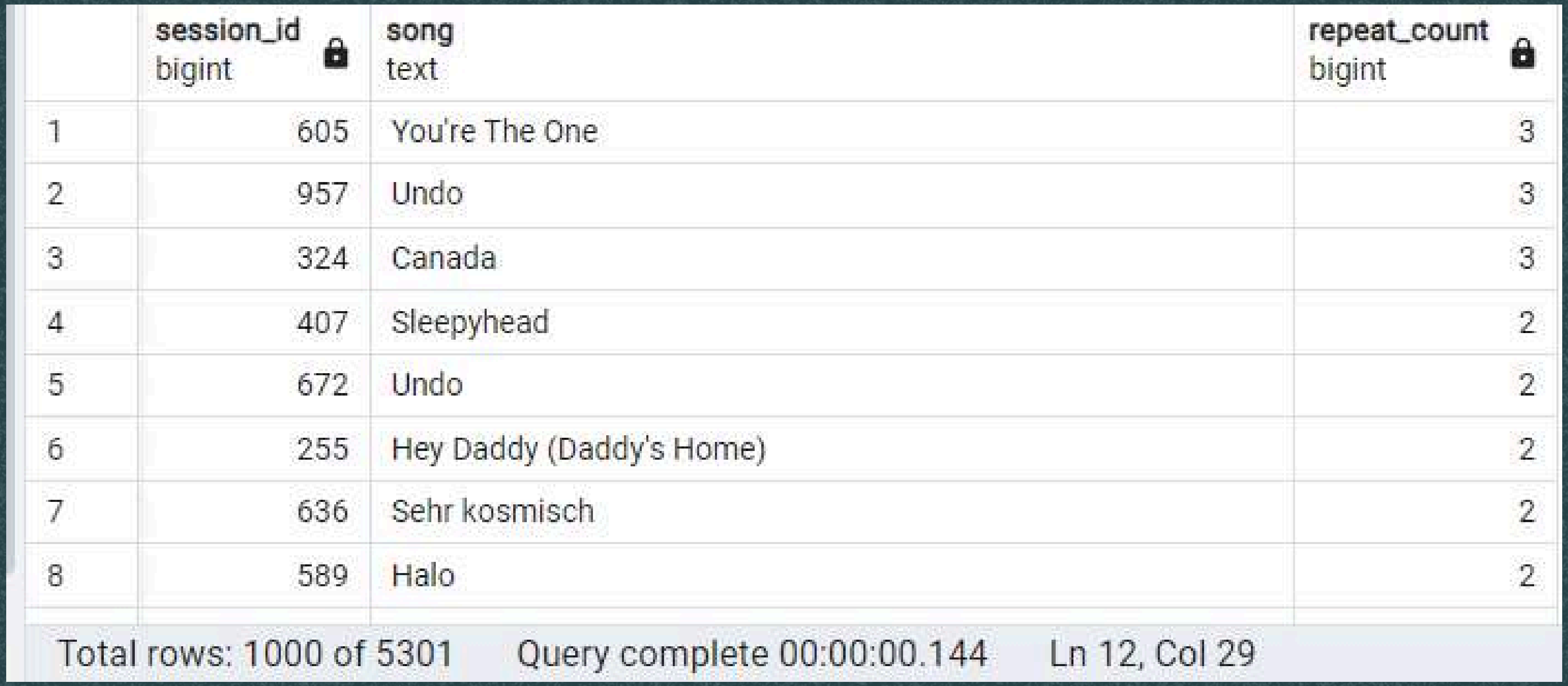
It helps in understanding user preferences, creating tailored playlists, optimizing content acquisition, and driving marketing campaigns.

User who have listened to same artist multiple time in a year

It helps in recognizing loyal fans, driving personalized recommendations, enhancing targeted marketing, and offering exclusive content.



```
-- Find the most playes song in a session
WITH my
AS
(
SELECT
    session_id,
    song,
    COUNT(song)
    AS Repeat_count,
    DENSE_RANK()
    OVER(PARTITION BY session_id
        ORDER BY COUNT(song) DESC)
        AS rank
FROM
    events
GROUP BY
    session_id, song)

SELECT
    session_id,
    song,
    Repeat_count
FROM my
WHERE rank = 1
ORDER BY
    Repeat_count DESC
```



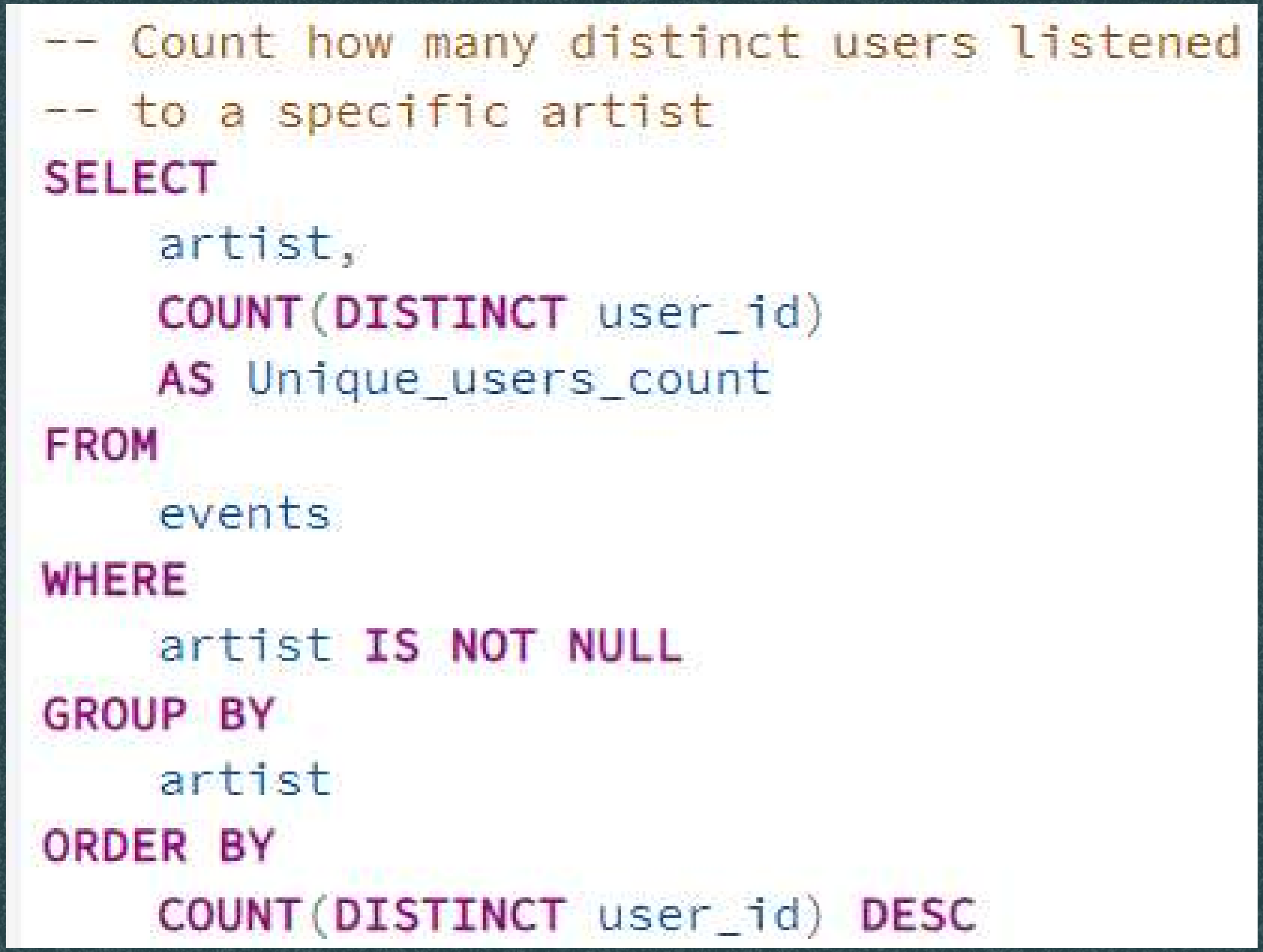
	session_id bigint	song text	repeat_count bigint
1	605	You're The One	3
2	957	Undo	3
3	324	Canada	3
4	407	Sleepyhead	2
5	672	Undo	2
6	255	Hey Daddy (Daddy's Home)	2
7	636	Sehr kosmisch	2
8	589	Halo	2

```
Total rows: 1000 of 5301     Query complete 00:00:00.144     Ln 12, Col 29
```

Find the most playes song in a session

It helps in understanding user engagement and user preference patterns, which can be used for targeted advertising.

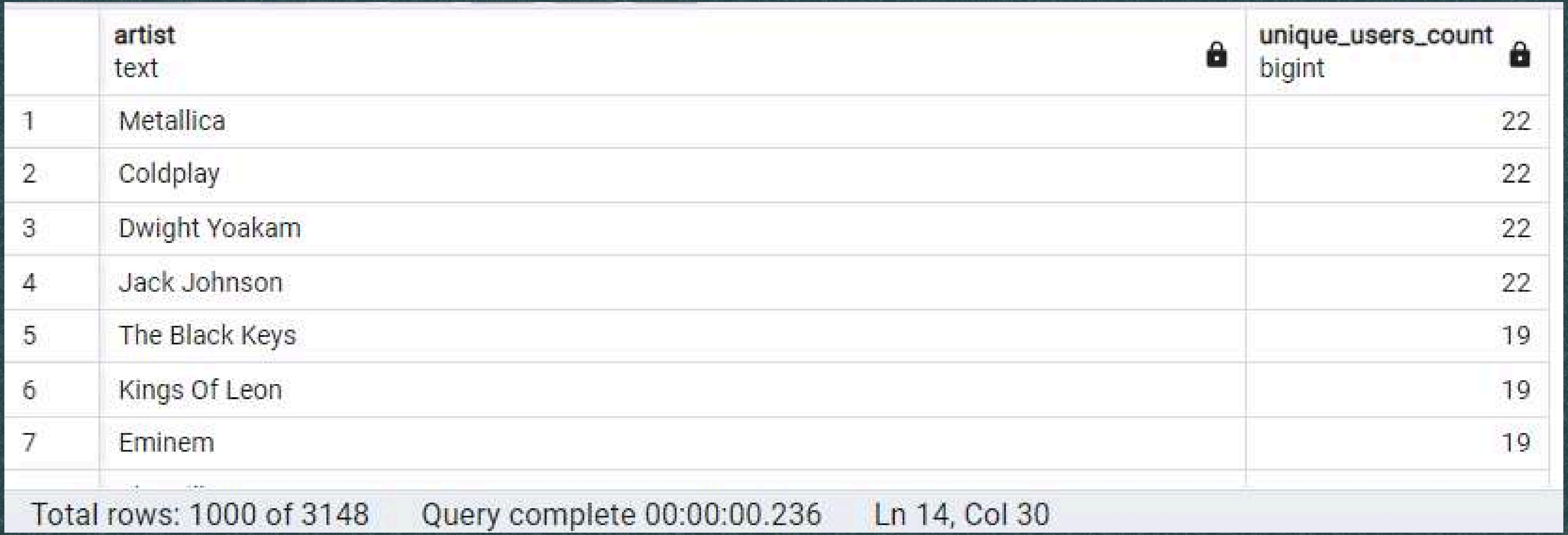
02



```
-- Count how many distinct users listened
-- to a specific artist
SELECT
    artist,
    COUNT(DISTINCT user_id)
    AS Unique_users_count
FROM
    events
WHERE
    artist IS NOT NULL
GROUP BY
    artist
ORDER BY
    COUNT(DISTINCT user_id) DESC
```

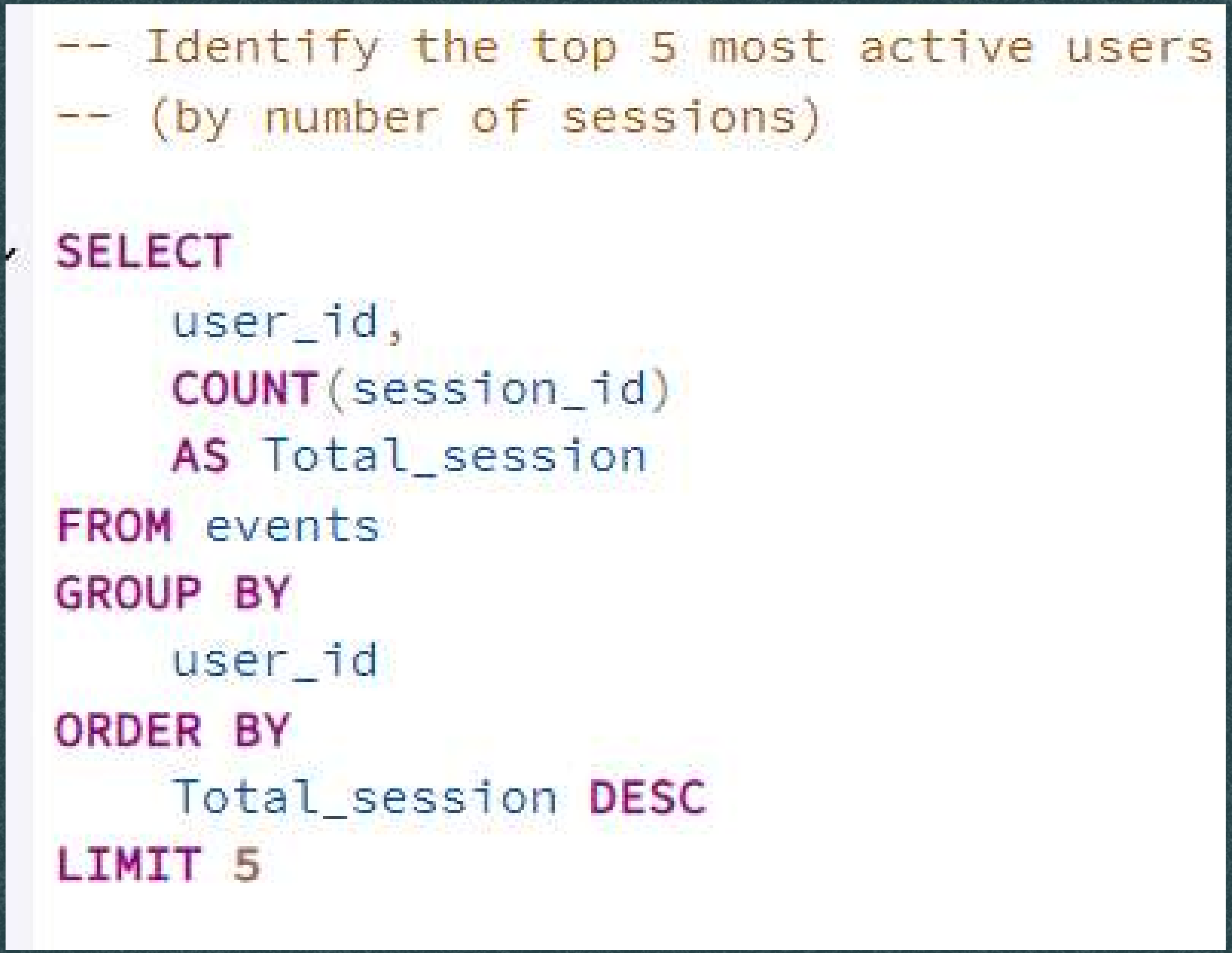
Count how many distinct users listened to a specific artist

It helps in Artist Popularity Measurement, which will be useful for making targeted Artist Campaigns and helps in Revenue Generation.

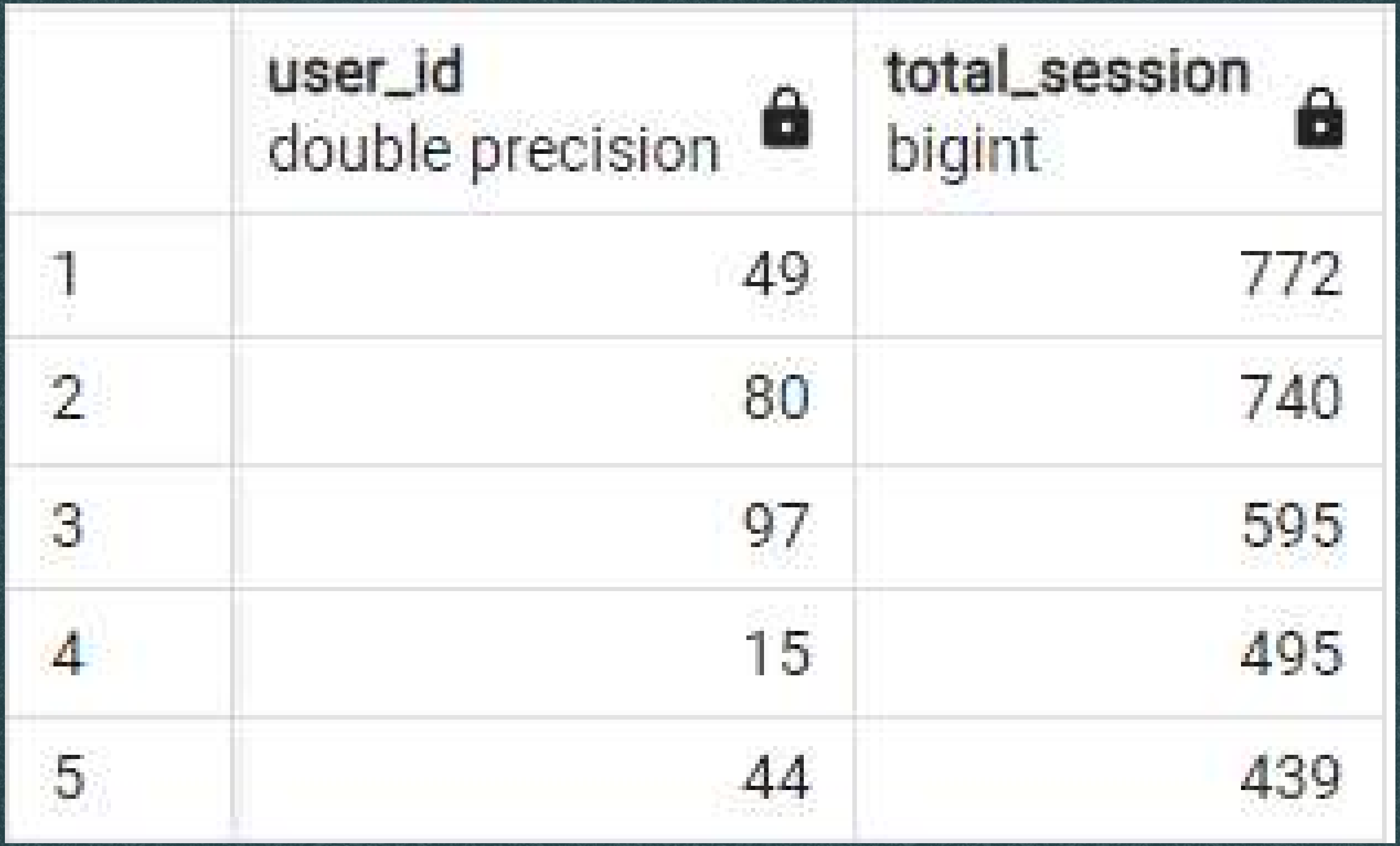


	artist text	unique_users_count bigint
1	Metallica	22
2	Coldplay	22
3	Dwight Yoakam	22
4	Jack Johnson	22
5	The Black Keys	19
6	Kings Of Leon	19
7	Eminem	19

Total rows: 1000 of 3148   Query complete 00:00:00.236   Ln 14, Col 30



```
-- Identify the top 5 most active users
-- (by number of sessions)

SELECT
    user_id,
    COUNT(session_id)
    AS Total_session
FROM events
GROUP BY
    user_id
ORDER BY
    Total_session DESC
LIMIT 5
```

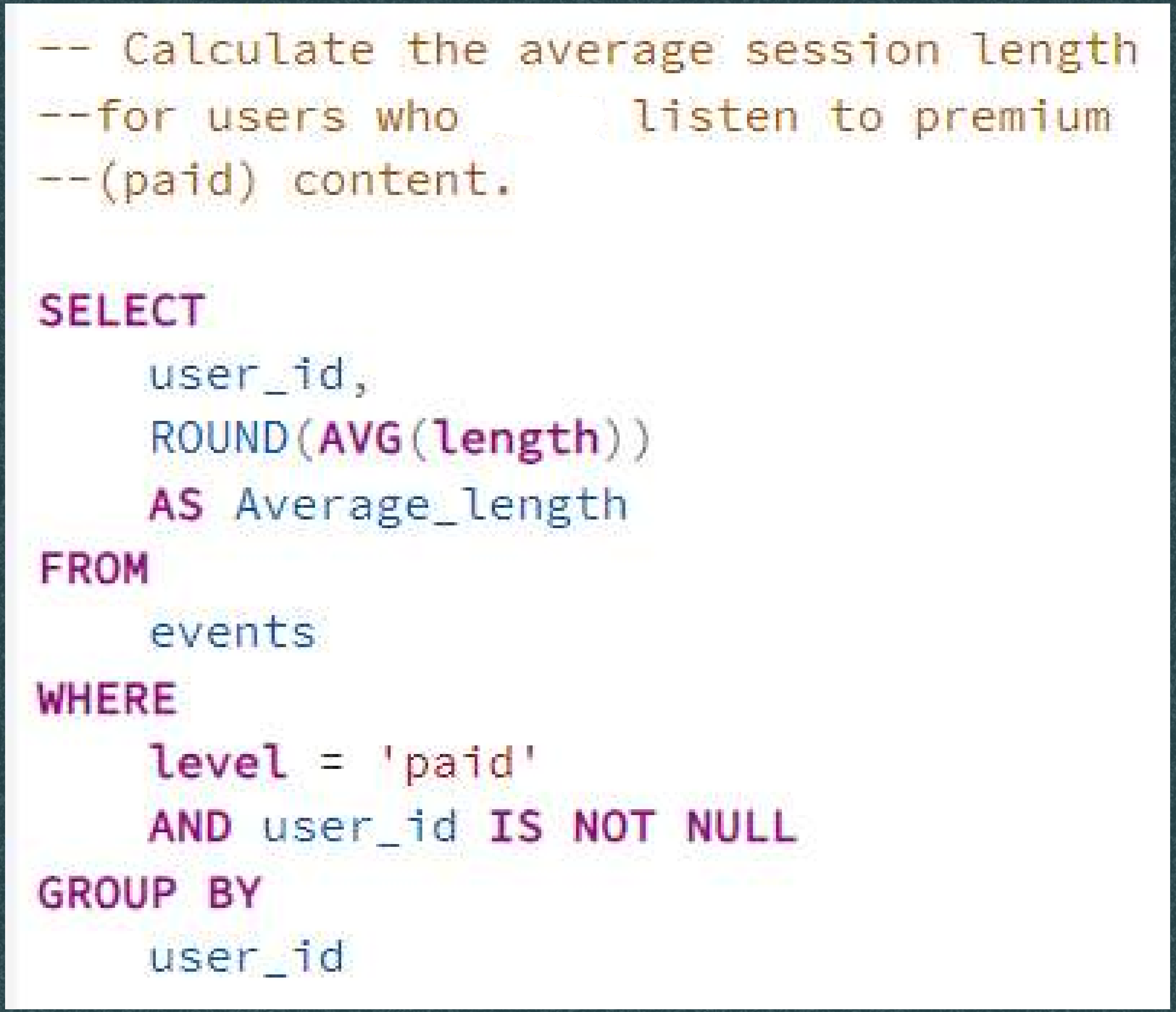


	user_id double precision	total_session bigint
1	49	772
2	80	740
3	97	595
4	15	495
5	44	439

Identify the top 5 most active users (by number of sessions)

We can do targeted marketing, promotions and personalized content recommendations by utilizing insights gained from understanding power users.

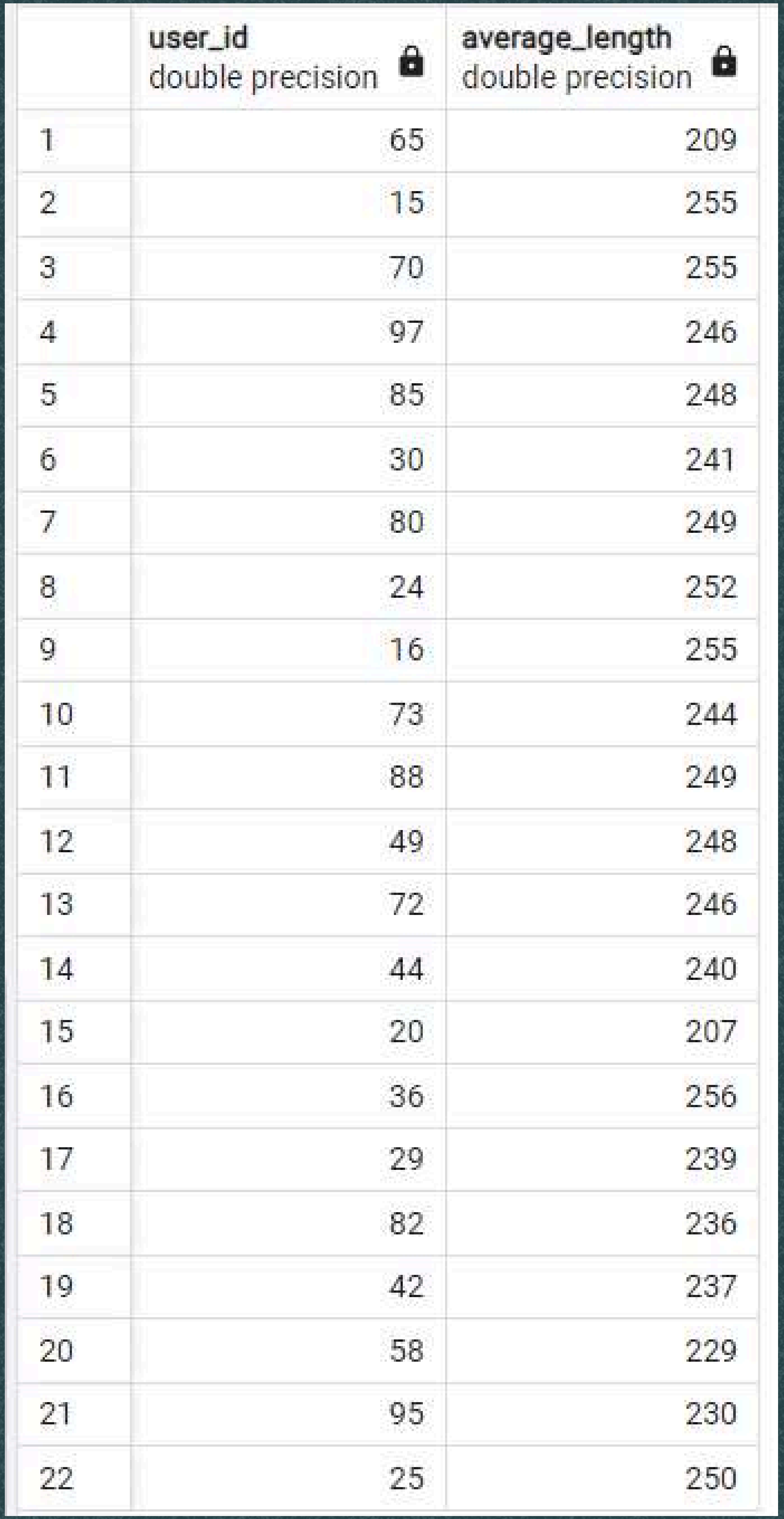
04



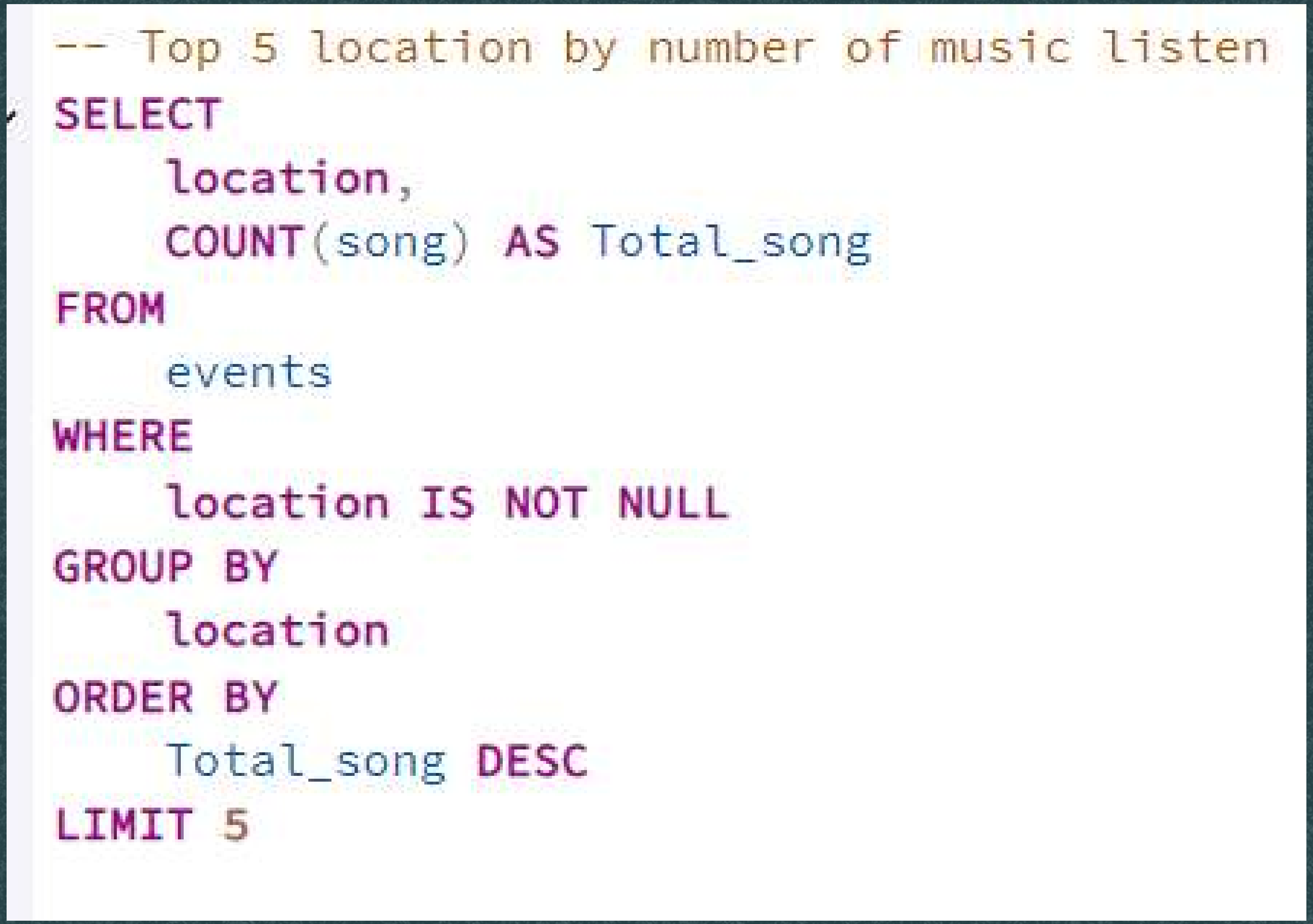
```
-- Calculate the average session length
--for users who     listen to premium
--(paid) content.

SELECT
    user_id,
    ROUND(AVG(length))
    AS Average_length
FROM
    events
WHERE
    level = 'paid'
    AND user_id IS NOT NULL
GROUP BY
    user_id
```

Calculate the average session length for users who listen to premium (paid) content

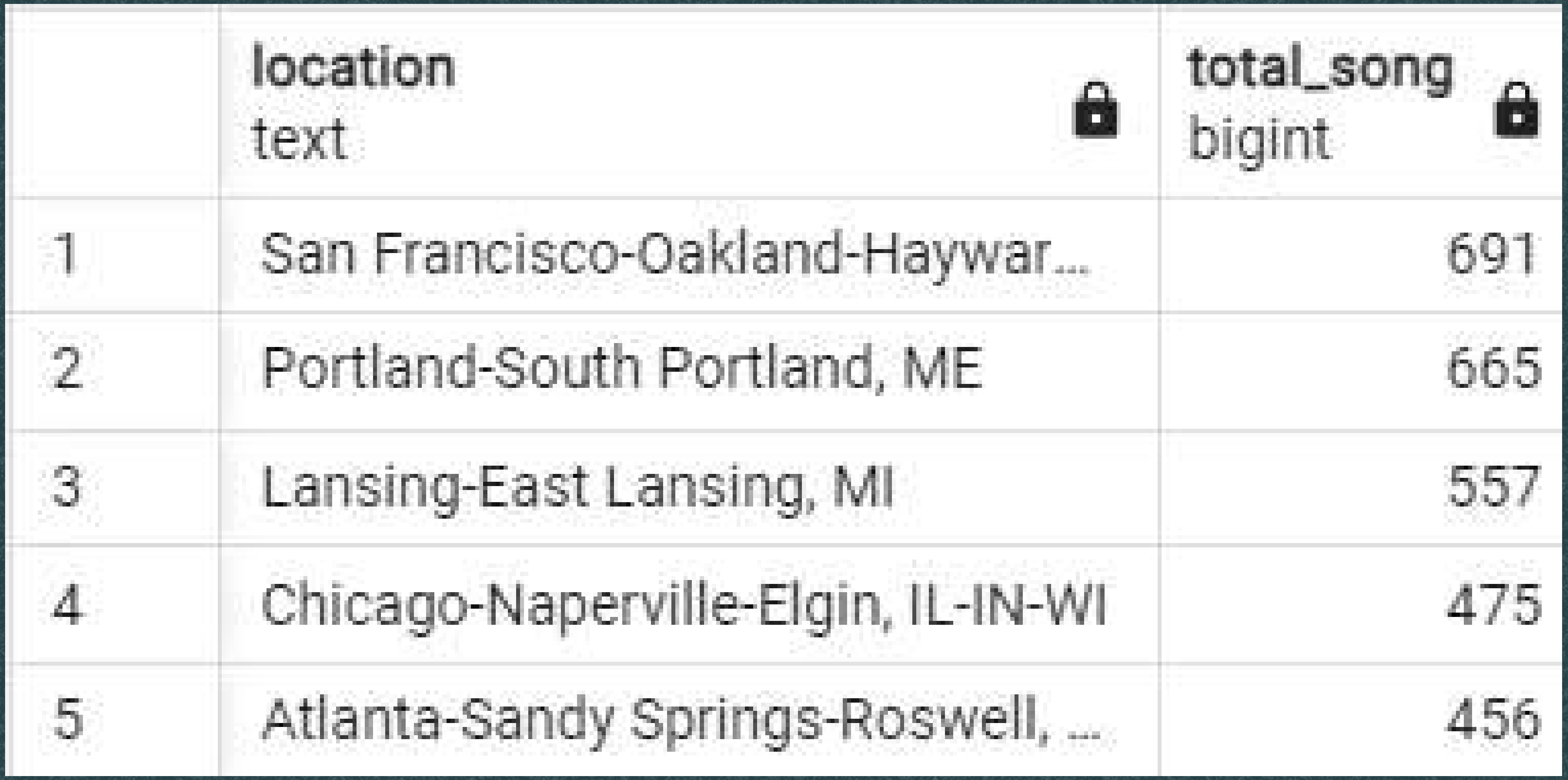
Understanding premium user engagement can support business decision-making, including revenue growth and cost-benefit analysis.



	user_id double precision	average_length double precision
1	65	209
2	15	255
3	70	255
4	97	246
5	85	248
6	30	241
7	80	249
8	24	252
9	16	255
10	73	244
11	88	249
12	49	248
13	72	246
14	44	240
15	20	207
16	36	256
17	29	239
18	82	236
19	42	237
20	58	229
21	95	230
22	25	250



```
-- Top 5 location by number of music listen
SELECT
    location,
    COUNT(song) AS Total_song
FROM
    events
WHERE
    location IS NOT NULL
GROUP BY
    location
ORDER BY
    Total_song DESC
LIMIT 5
```



	location text	total_song bigint
1	San Francisco-Oakland-Haywar...	691
2	Portland-South Portland, ME	665
3	Lansing-East Lansing, MI	557
4	Chicago-Naperville-Elgin, IL-IN-WI	475
5	Atlanta-Sandy Springs-Roswell, ...	456

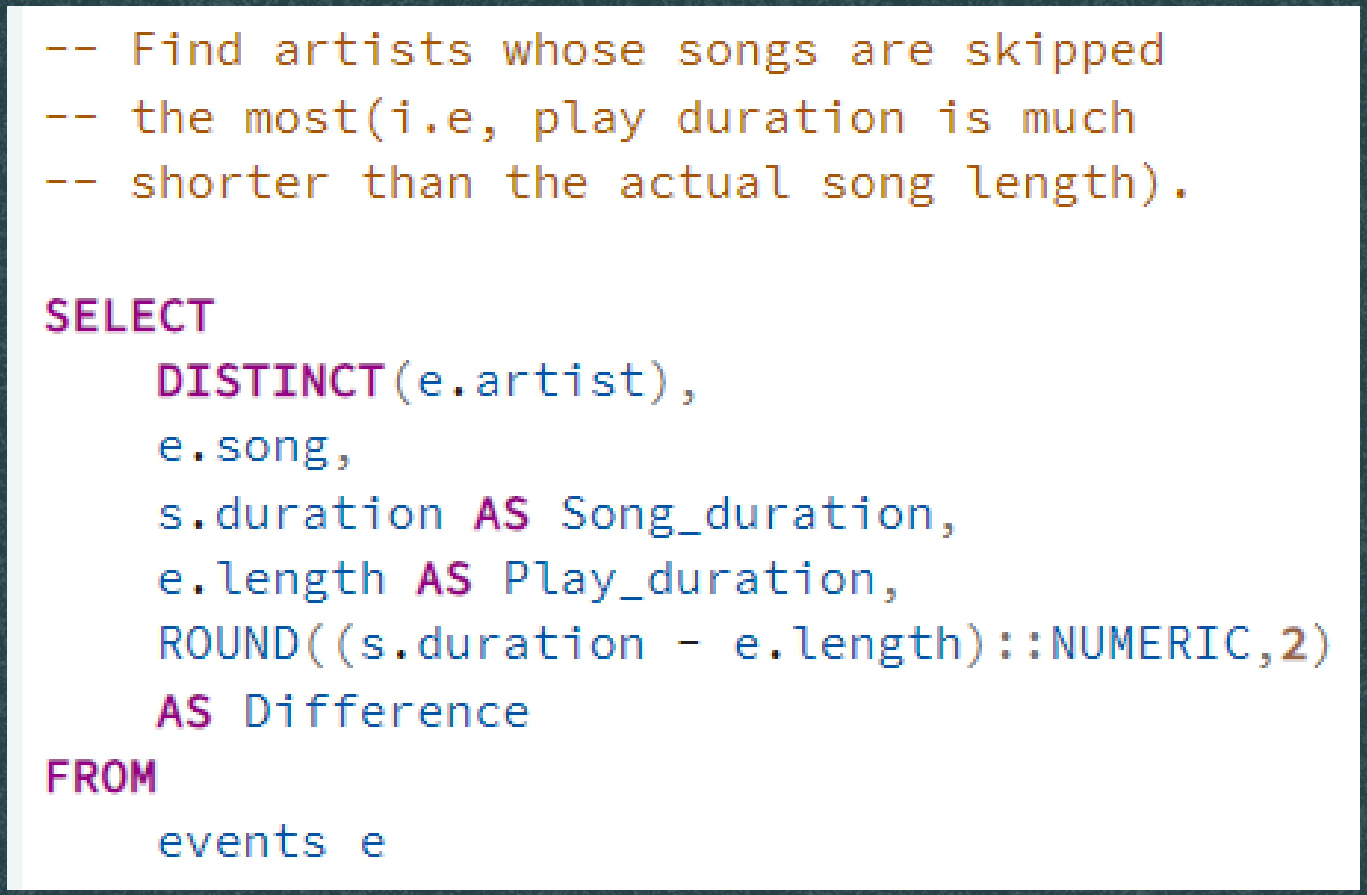
<u>Top 5 location by number of music listen</u>

It helps the platform understand regional preferences, enabling targeted marketing, enhancing content strategy, optimizing user experience and informing business decisions.

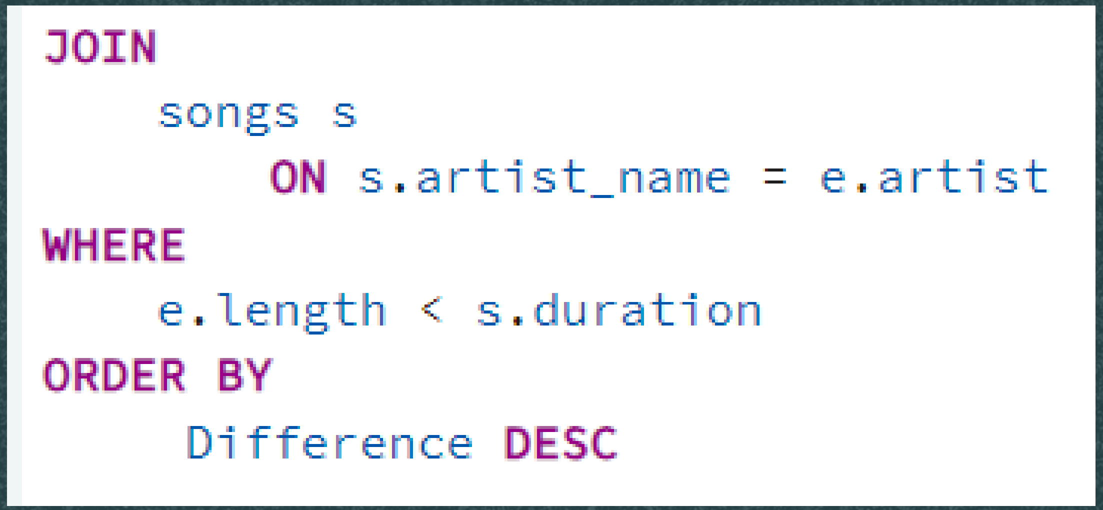
06

Find artists whose songs are skipped the most

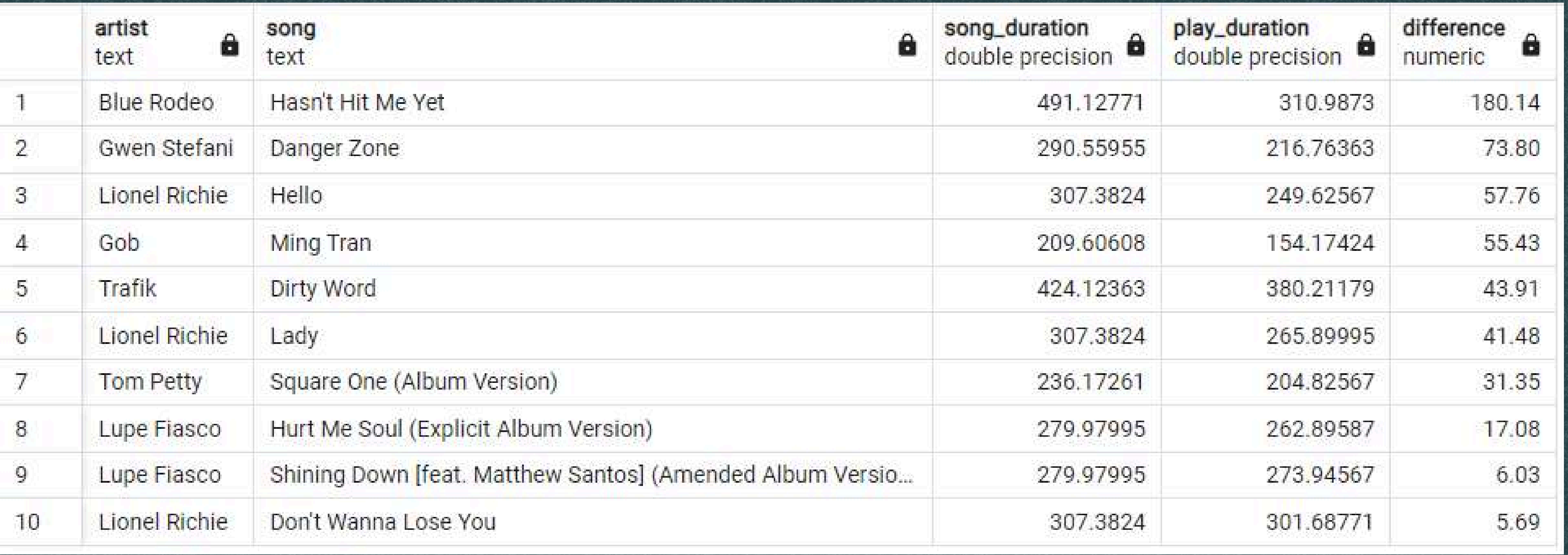
It helps in assessing content quality, improving user experience, providing feedback to artists, optimizing recommendation algorithms, and guiding advertising and revenue strategies.



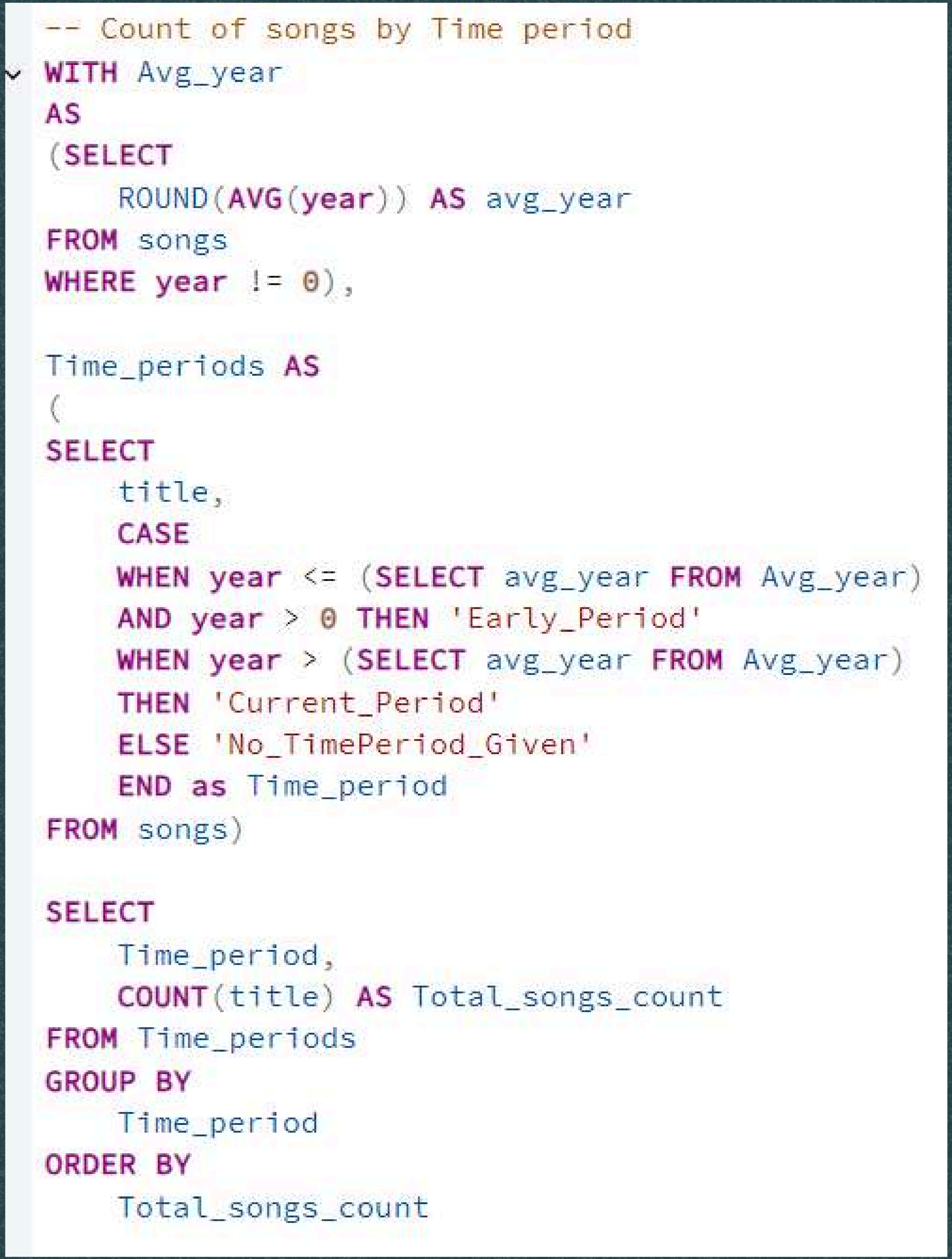
```
-- Find artists whose songs are skipped
-- the most(i.e, play duration is much
-- shorter than the actual song length).

SELECT
    DISTINCT(e.artist),
    e.song,
    s.duration AS Song_duration,
    e.length AS Play_duration,
    ROUND((s.duration - e.length)::NUMERIC,2)
    AS Difference
FROM
    events e
```



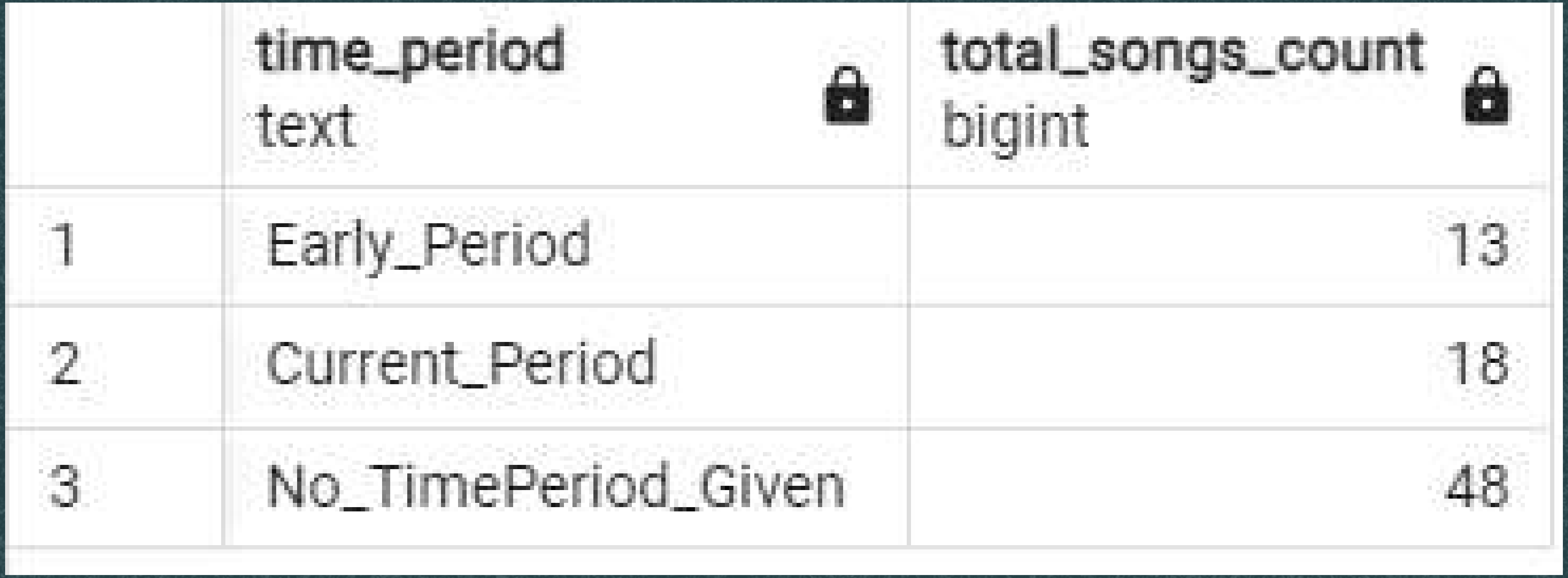
```
JOIN
    songs s
        ON s.artist_name = e.artist
WHERE
    e.length < s.duration
ORDER BY
     Difference DESC
```



	artist text	song text	song_duration double precision	play_duration double precision	difference numeric
1	Blue Rodeo	Hasn't Hit Me Yet	491.12771	310.9873	180.14
2	Gwen Stefani	Danger Zone	290.55955	216.76363	73.80
3	Lionel Richie	Hello	307.3824	249.62567	57.76
4	Gob	Ming Tran	209.60608	154.17424	55.43
5	Trafik	Dirty Word	424.12363	380.21179	43.91
6	Lionel Richie	Lady	307.3824	265.89995	41.48
7	Tom Petty	Square One (Album Version)	236.17261	204.82567	31.35
8	Lupe Fiasco	Hurt Me Soul (Explicit Album Version)	279.97995	262.89587	17.08
9	Lupe Fiasco	Shining Down [feat. Matthew Santos] (Amended Album Versio...	279.97995	273.94567	6.03
10	Lionel Richie	Don't Wanna Lose You	307.3824	301.68771	5.69



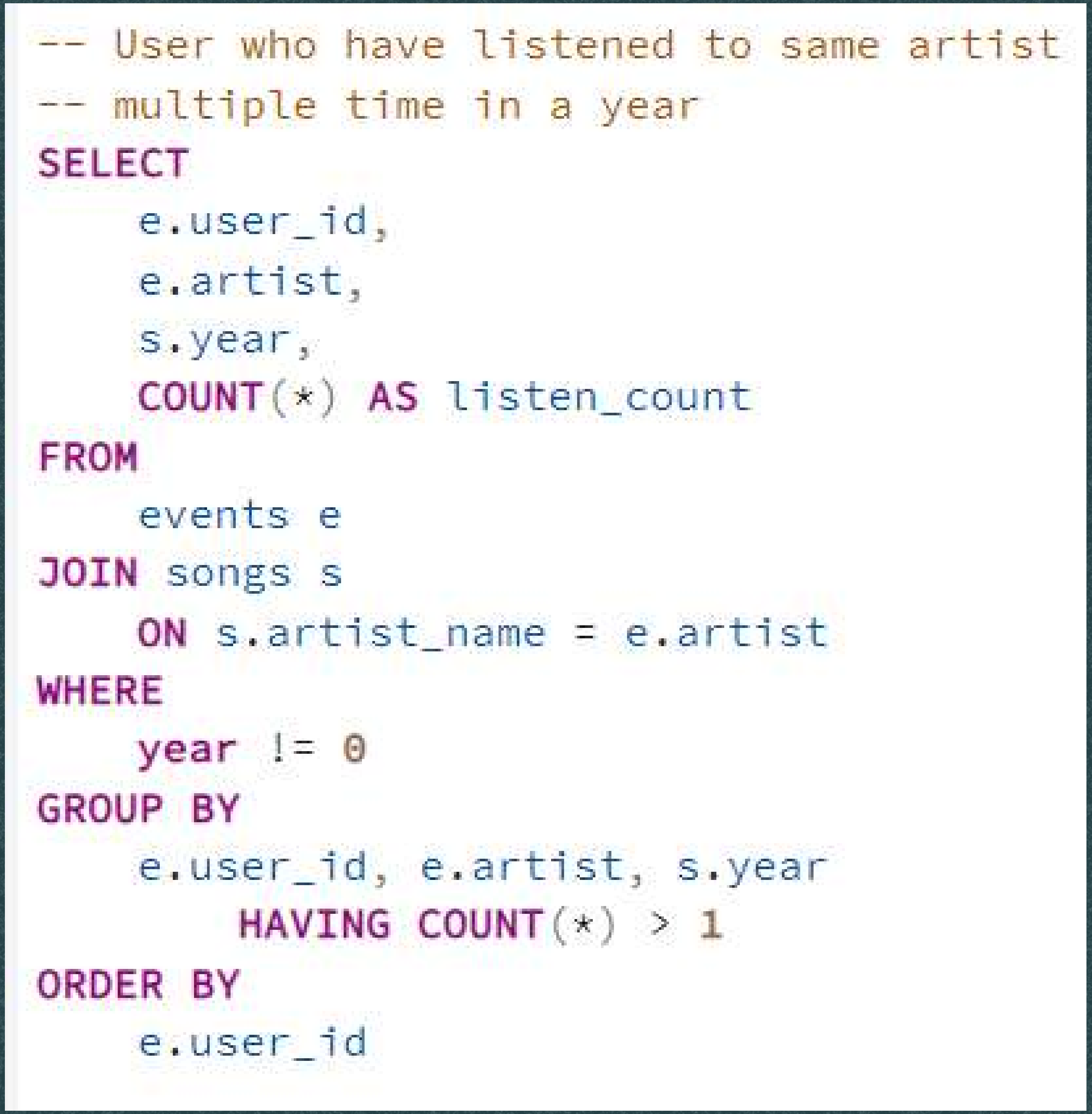
```
-- Count of songs by Time period
WITH Avg_year
AS
(SELECT
    ROUND(AVG(year)) AS avg_year
FROM songs
WHERE year != 0),

Time_periods AS
(
SELECT
    title,
    CASE
    WHEN year <= (SELECT avg_year FROM Avg_year)
    AND year > 0 THEN 'Early_Period'
    WHEN year > (SELECT avg_year FROM Avg_year)
    THEN 'Current_Period'
    ELSE 'No_TimePeriod_Given'
    END as Time_period
FROM songs)

SELECT
    Time_period,
    COUNT(title) AS Total_songs_count
FROM Time_periods
GROUP BY
    Time_period
ORDER BY
    Total_songs_count
```



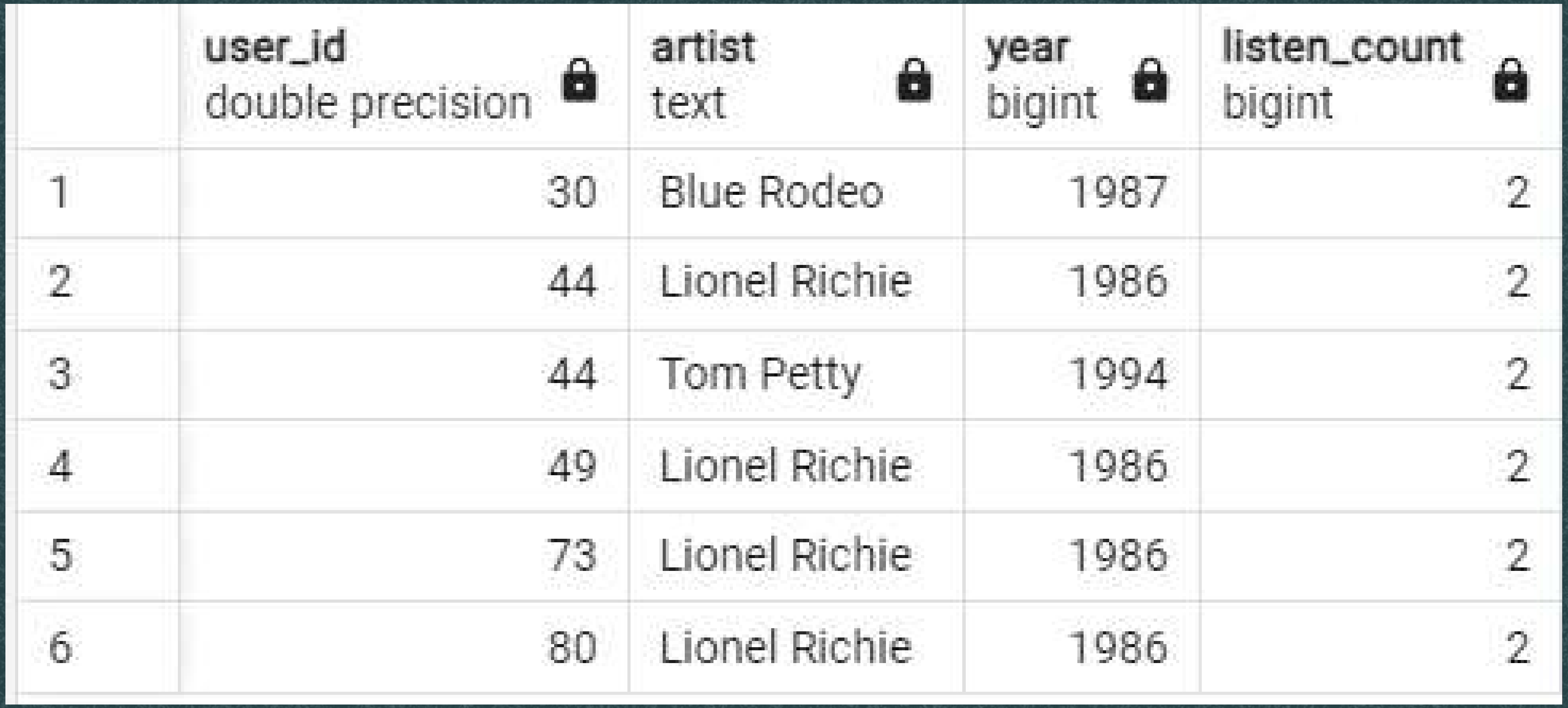
	time_period text	total_songs_count bigint
1	Early_Period	13
2	Current_Period	18
3	No_TimePeriod_Given	48

Count of songs by Time period

It helps in understanding user preferences, creating tailored playlists, optimizing content acquisition, and driving marketing campaigns.



```
-- User who have listened to same artist
-- multiple time in a year
SELECT
    e.user_id,
    e.artist,
    s.year,
    COUNT(*) AS listen_count
FROM
    events e
JOIN songs s
    ON s.artist_name = e.artist
WHERE
    year != 0
GROUP BY
    e.user_id, e.artist, s.year
        HAVING COUNT(*) > 1
ORDER BY
    e.user_id
```



	user_id double precision	artist text	year bigint	listen_count bigint
1	30	Blue Rodeo	1987	2
2	44	Lionel Richie	1986	2
3	44	Tom Petty	1994	2
4	49	Lionel Richie	1986	2
5	73	Lionel Richie	1986	2
6	80	Lionel Richie	1986	2

User who have listened to same artist multiple time in a year

It helps in recognizing loyal fans, driving personalized recommendations, enhancing targeted marketing, and offering exclusive content.

THANK YOU

Yuvraj Giri