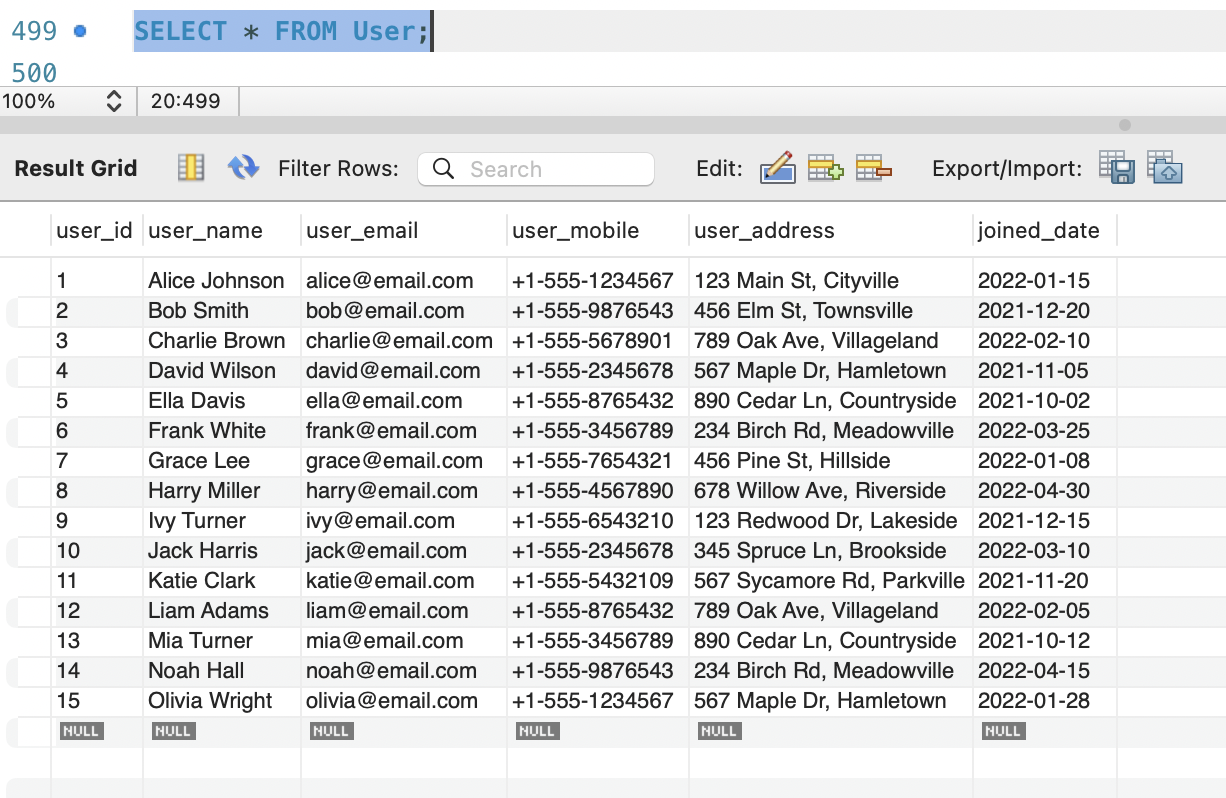
**Project Deliverable-3**

**Query 1**: Select All Users  
  
**Description**: Retrieve all user records from the "User" table.  
  
**SQL Statement:**  
  
SELECT \* FROM User;

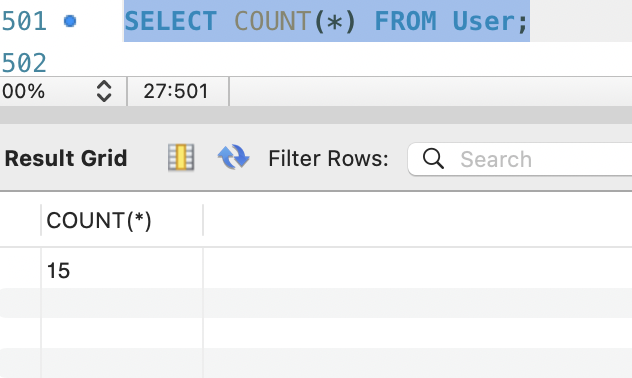


Query 2: Count the Number of Users

Description: Count the total number of users in the "User" table.

**SQL Statement:**

SELECT COUNT(\*) FROM User;



**Query 3:** List Songs by Rating

**Description:** Retrieve song names and their ratings, ordered by rating in descending order.

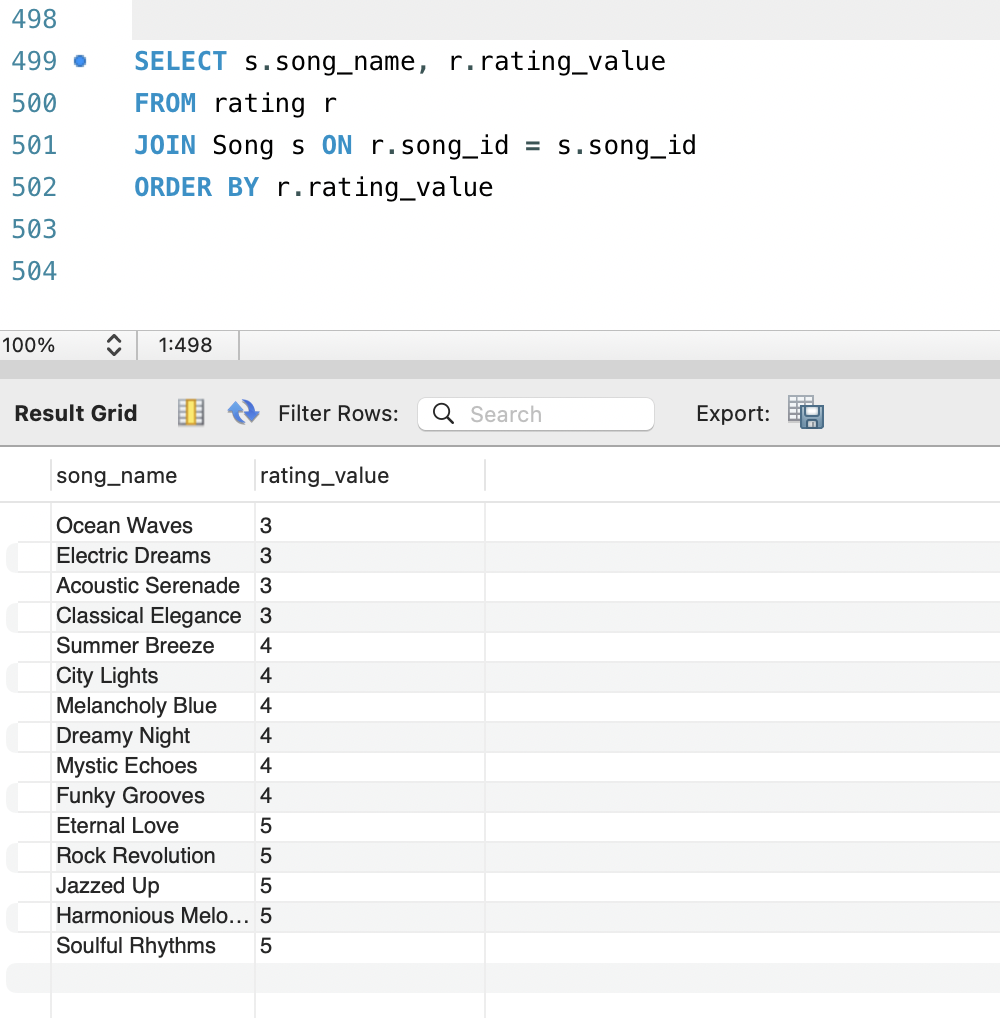
**SQL Statement:**

SELECT s.song\_name, r.rating\_value

FROM rating r

JOIN Song s ON r.song\_id = s.song\_id

ORDER BY r.rating\_value



**Query 4:** Find Artists with More than One Album

**Description:** Retrieve artists who have more than one album.

**SQL Statement:**

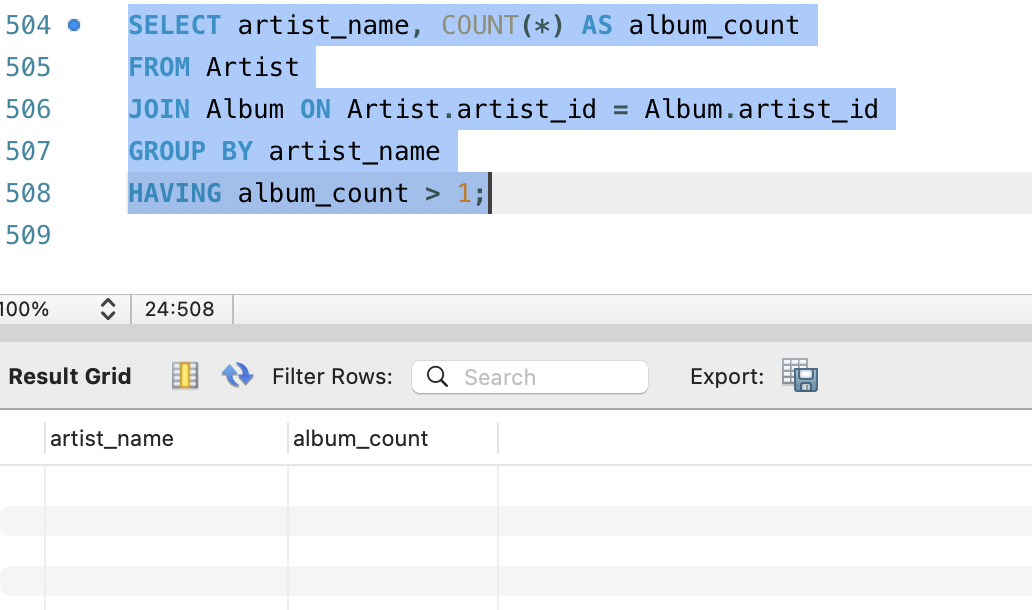
SELECT artist\_name, COUNT(\*) AS album\_count

FROM Artist

JOIN Album ON Artist.artist\_id = Album.artist\_id

GROUP BY artist\_name

HAVING album\_count > 1;



Since in the given data we only have an artist with single album the result is null.

**Query 5:** Window Function - Calculate Running Total of Song Ratings

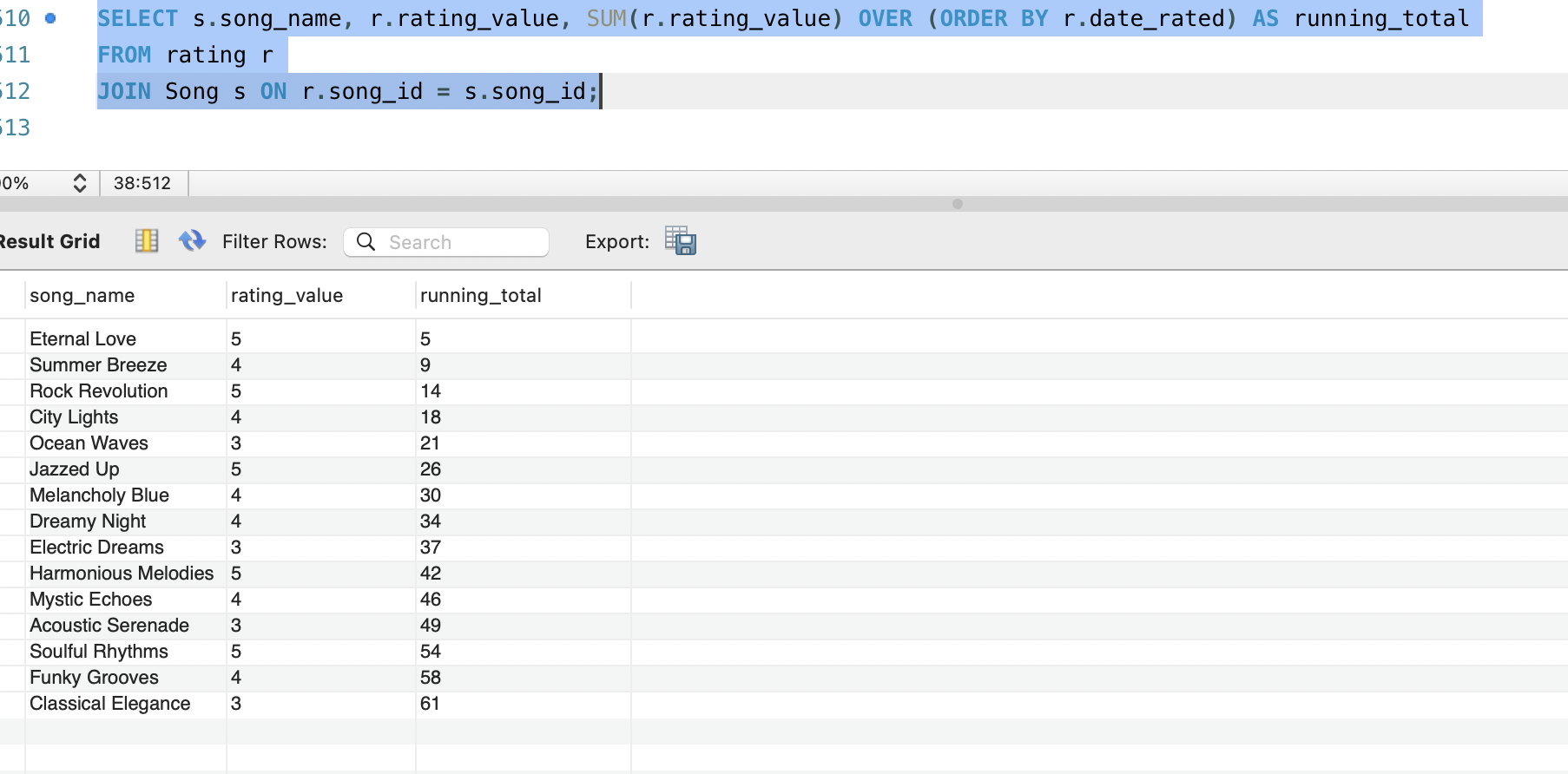
**Description:** Calculate the running total of song ratings for each song.

**SQL Statement:**

SELECT s.song\_name, r.rating\_value, SUM(r.rating\_value) OVER (ORDER BY r.date\_rated) AS running\_total

FROM rating r

JOIN Song s ON r.song\_id = s.song\_id;



**Query 6:** Find Top 5 Users by Joined Date

**Description:** Retrieve the top 5 users who joined the system most recently.

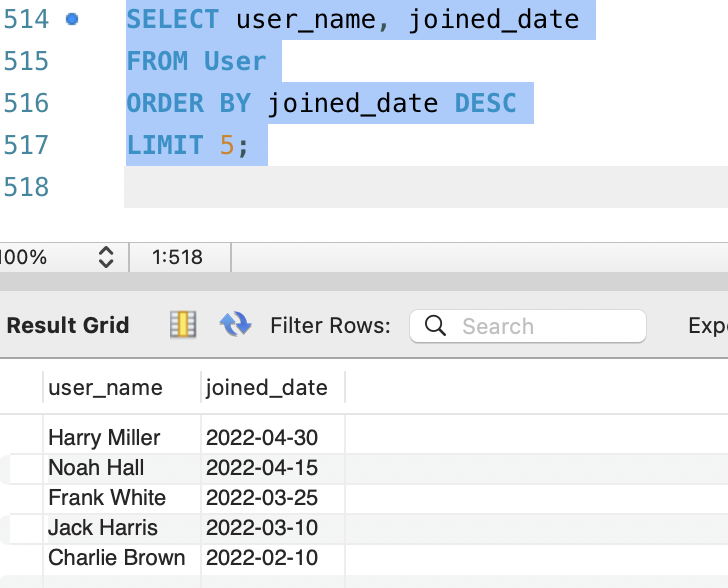
**SQL Statement:**

SELECT user\_name, joined\_date

FROM User

ORDER BY joined\_date DESC

LIMIT 5;



**Query 7:** Calculate Average Song Rating by Language

**Description:** Calculate the average song rating for each language.

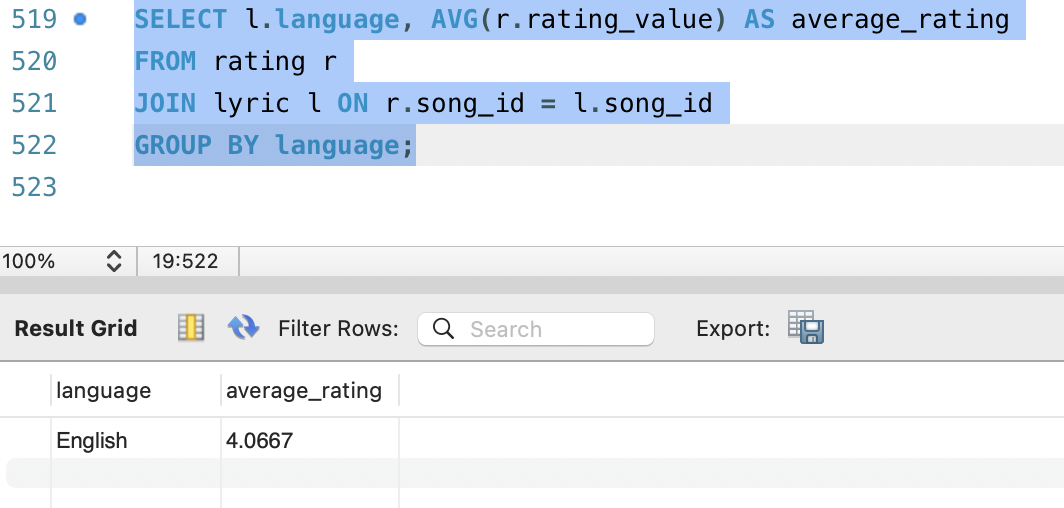
**SQL Statement:**

SELECT l.language, AVG(r.rating\_value) AS average\_rating

FROM rating r

JOIN lyric l ON r.song\_id = l.song\_id

GROUP BY language;



**Query 8:** OLAP Query - Find Total Sales by Album Type

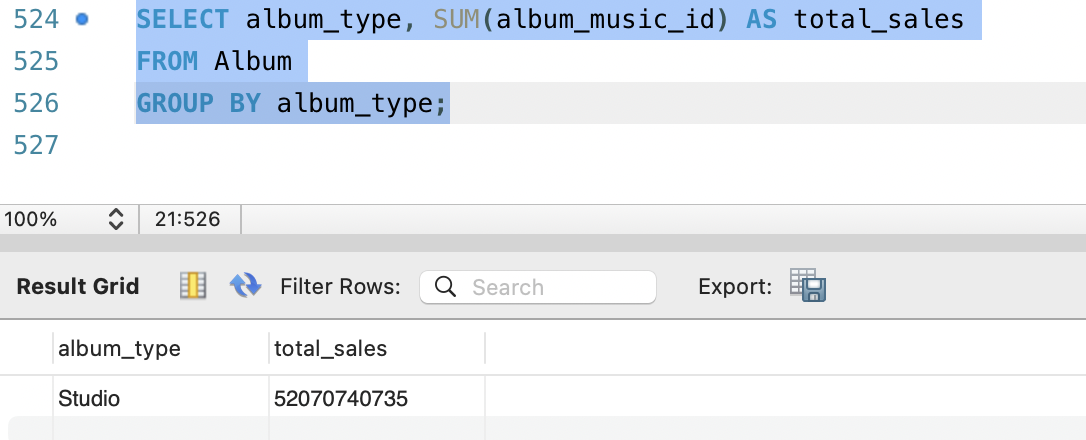
**Description:** Calculate the total sales for each album type.

**SQL Statement:**

SELECT album\_type, SUM(album\_music\_id) AS total\_sales

FROM Album

GROUP BY album\_type;



**Query 9:** Calculate the Total Chords Created by Each User

**Description:** Calculate the total number of chords created by each user.

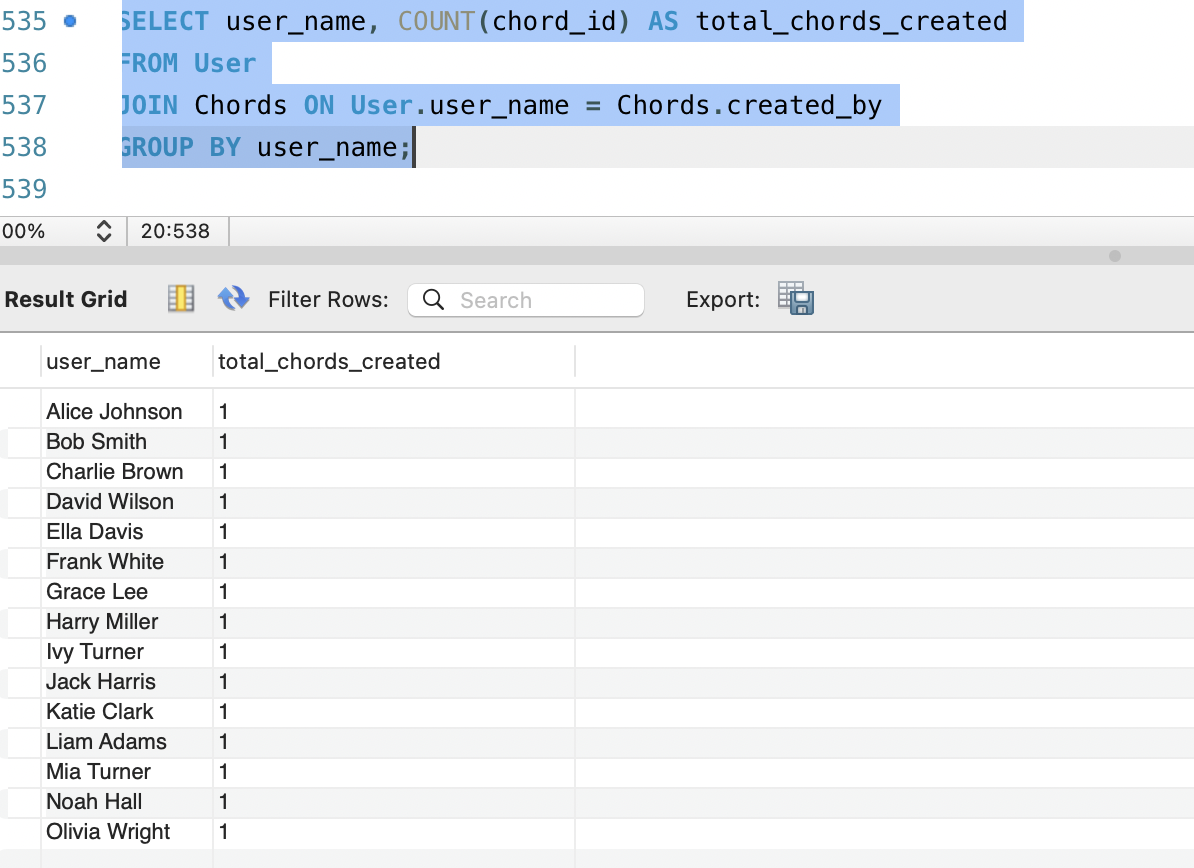
**SQL Statement:**

SELECT user\_name, COUNT(chord\_id) AS total\_chords\_created

FROM User

JOIN Chord ON User.user\_name = Chord.created\_by

GROUP BY user\_name;



**Query 10:** Window Function - Rank Songs by Rating

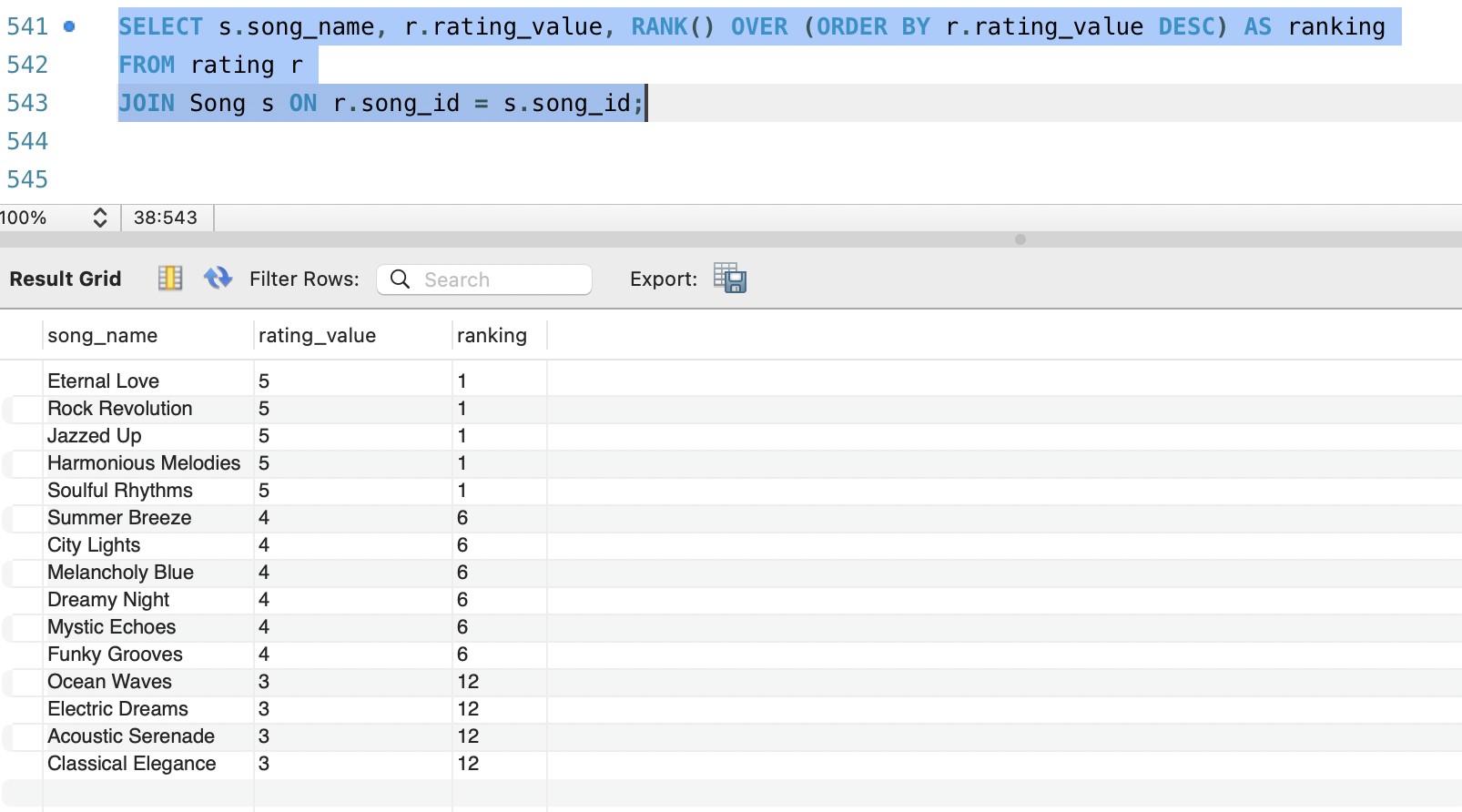
**Description:** Rank songs by their ratings.

**SQL Statement:**

SELECT s.song\_name, r.rating\_value, RANK() OVER (ORDER BY r.rating\_value DESC) AS ranking

FROM rating r

JOIN Song s ON r.song\_id = s.song\_id;



**Query 11:** Find the User with the Most Playlists

**Description:** Retrieve the user with the most playlists.

**SQL Statement:**

SELECT user\_name, COUNT(playlist\_id) AS playlist\_count

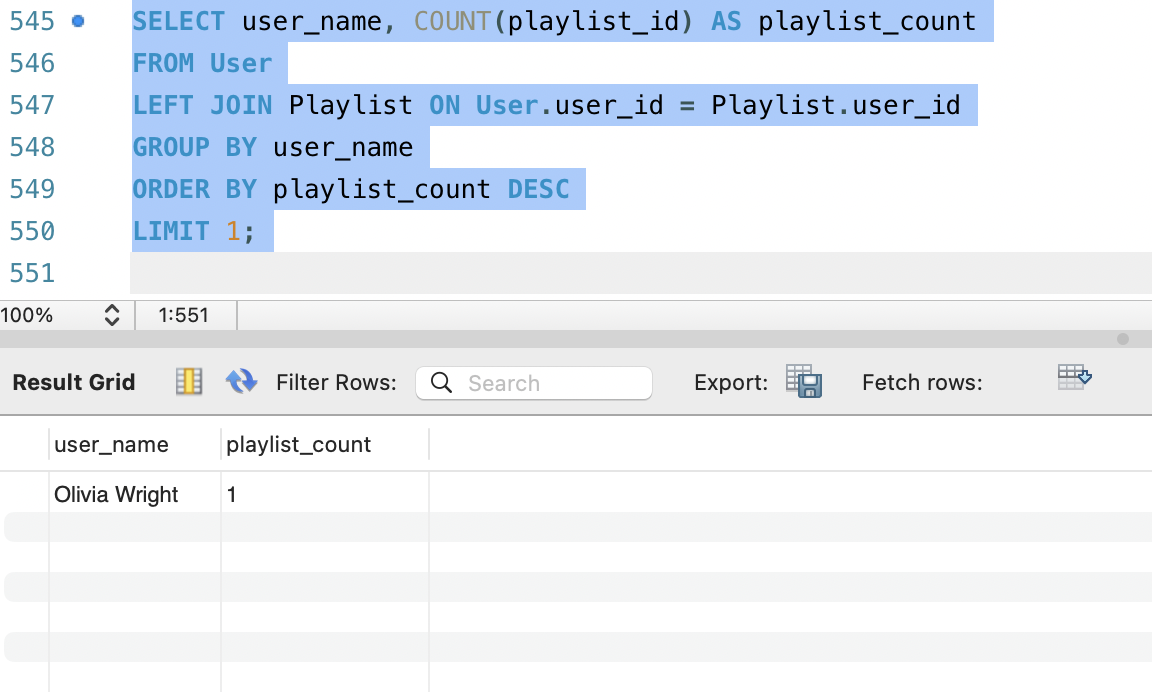
FROM User

LEFT JOIN Playlist ON User.user\_id = Playlist.user\_id

GROUP BY user\_name

ORDER BY playlist\_count DESC

LIMIT 1;



**Query 12:** Calculate Average Ratings by Artist Gender

**Description:** Calculate the average song ratings for each artist gender.

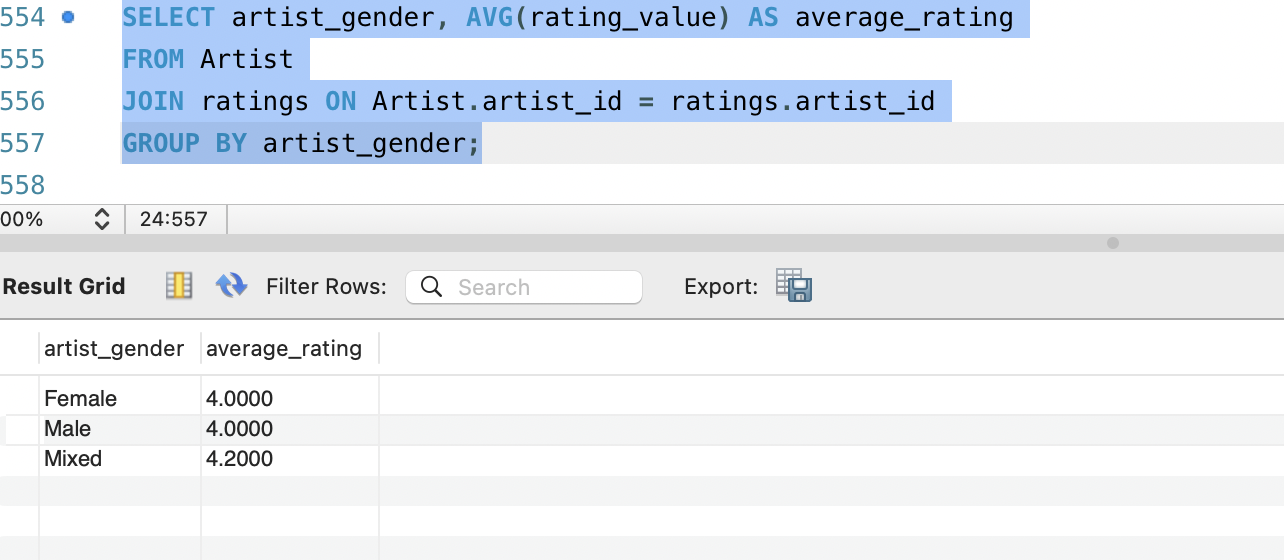
**SQL Statement:**

SELECT artist\_gender, AVG(rating\_value) AS average\_rating

FROM Artist

JOIN ratings ON Artist.artist\_id = ratings.artist\_id

GROUP BY artist\_gender;



**Query 13:** OLAP Query - Calculate Total Ratings for Each Song Category

**Description:** Calculate the total ratings for each song category.

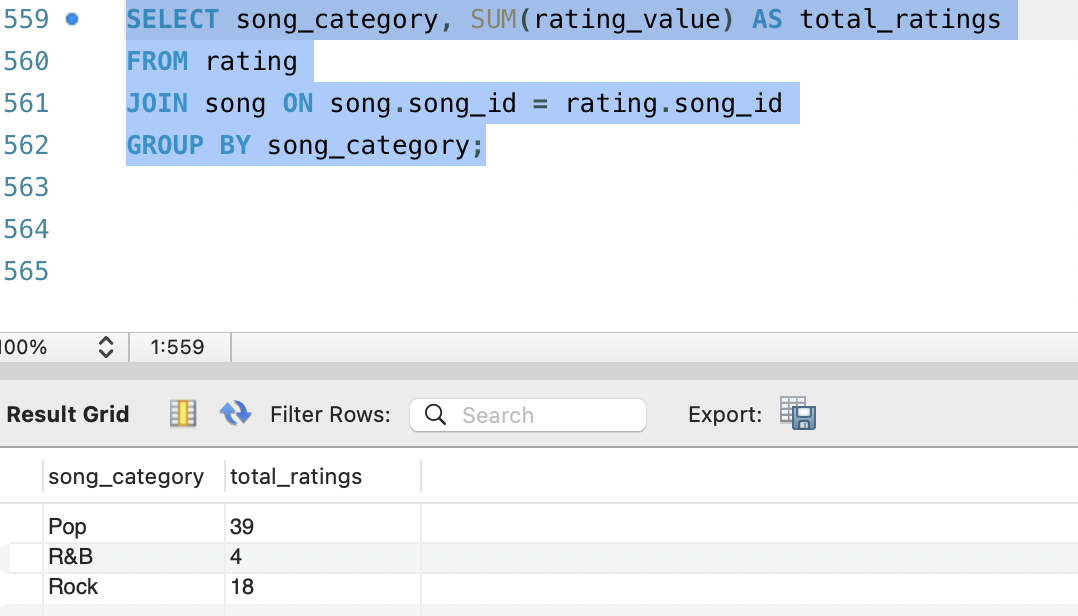
**SQL Statement:**

SELECT song\_category, SUM(rating\_value) AS total\_ratings

FROM rating

JOIN song ON song.song\_id = rating.song\_id

GROUP BY song\_category;



**Query 14:** Window Function - Find the Artist with the Most Songs

**Description:** Find the artist with the most songs and their count.

**SQL Statement:**

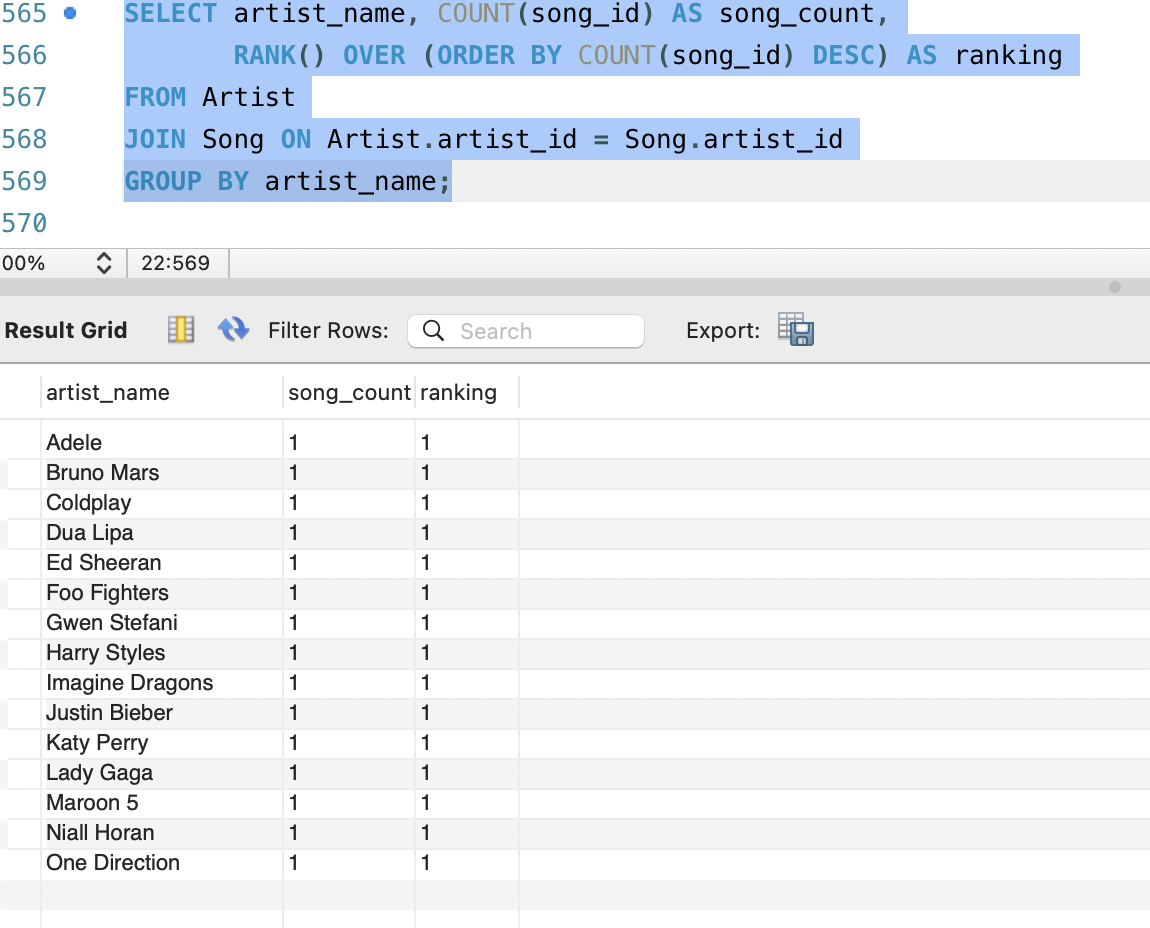
SELECT artist\_name, COUNT(song\_id) AS song\_count,

RANK() OVER (ORDER BY COUNT(song\_id) DESC) AS ranking

FROM Artist

JOIN Song ON Artist.artist\_id = Song.artist\_id

GROUP BY artist\_name;



**Query 15:** Retrieve Songs and Their Lyric Language   
**Description:** Get a list of songs along with the language of their lyrics.   
**SQL Statement:**SELECT s.song\_name, l.language

FROM Song s

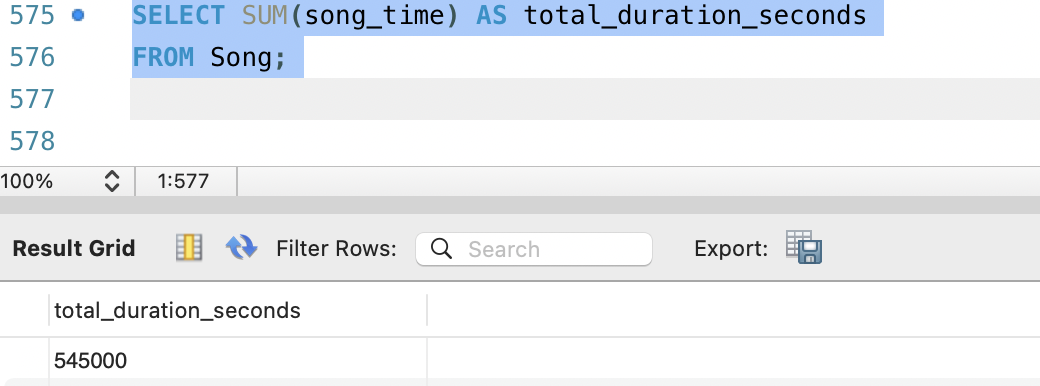
JOIN Lyric l ON s.song\_id = l.song\_id;



**Query 16:** Calculate the Total Duration of All Songs   
**Description:** Calculate the total duration (in seconds) of all songs.   
**SQL Statement:**

SELECT SUM(song\_time) AS total\_duration\_seconds

FROM Song;



**Query 17:** Find the Top 5 Artists with the Highest Average Song Ratings

**Description:** Identify the top 5 artists with the highest average song ratings.

**SQL Statement:**

SELECT a.artist\_name, AVG(s.song\_rating) AS average\_rating

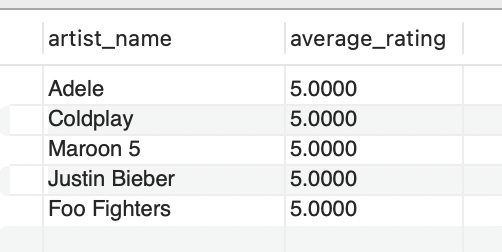
FROM Artist a

JOIN Song s ON a.artist\_id = s.artist\_id

GROUP BY a.artist\_name

ORDER BY average\_rating DESC

LIMIT 5;



**Query 18:** Calculate the Total Number of Chords Created by Each Artist

**Description:** Calculate the total number of chords created by each artist.

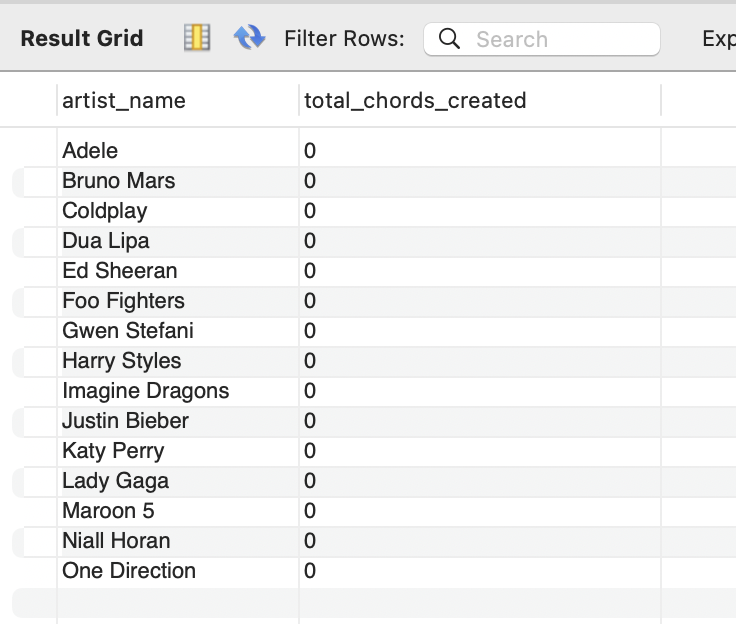
**SQL Statement:**

SELECT a.artist\_name, COUNT(c.chord\_id) AS total\_chords\_created

FROM Artist a

LEFT JOIN Chord c ON a.artist\_id = c.created\_by

GROUP BY a.artist\_name;



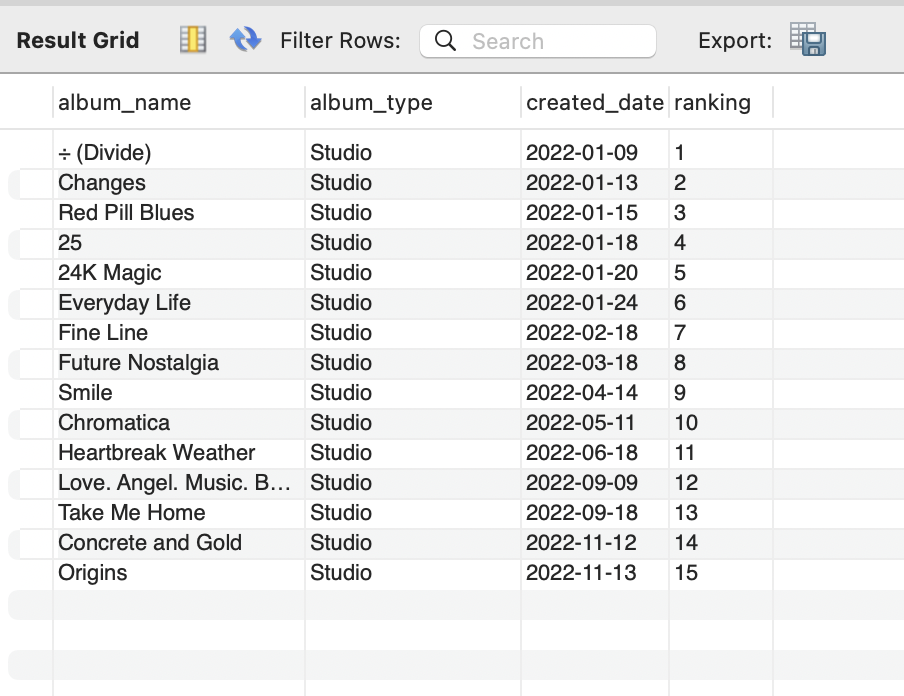
**Query 19:** Window Function - Rank Albums by Release Date Within Each Album Type **Description:** Rank albums by their release date within each album type.

**SQL Statement:**

SELECT album\_name, album\_type, created\_date,

RANK() OVER (PARTITION BY album\_type ORDER BY created\_date) AS ranking

FROM Album;



**Query 20:** Find Songs with the Same Name in Different Languages

**Description:** Identify songs with the same name but different lyrics languages.

**SQL Statement:**

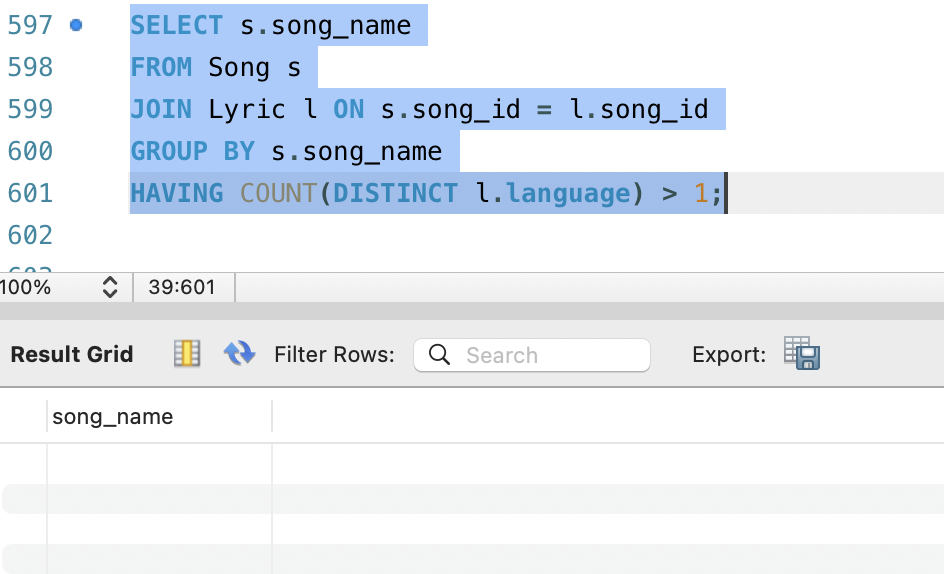
SELECT s.song\_name

FROM Song s

JOIN Lyric l ON s.song\_id = l.song\_id

GROUP BY s.song\_name

HAVING COUNT(DISTINCT l.language) > 1;



Since in the given data we only have english language we got null value.

**Query 21:** Calculate the Average Rating for Each User's Rated Songs

**Description:** Calculate the average rating for each user's rated songs.

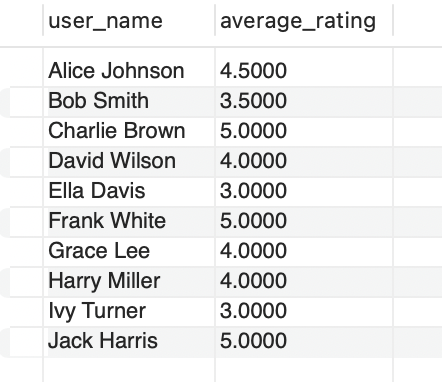
**SQL Statement:**

SELECT u.user\_name, AVG(r.rating\_value) AS average\_rating

FROM User u

JOIN Rating r ON u.user\_id = r.user\_id

GROUP BY u.user\_name;



**Query 22:** Find the Most Popular Genre

**Description:** Determine the most popular genre based on the number of songs.

**SQL Statement:**

SELECT g.gener\_name, COUNT(s.song\_id) AS song\_count

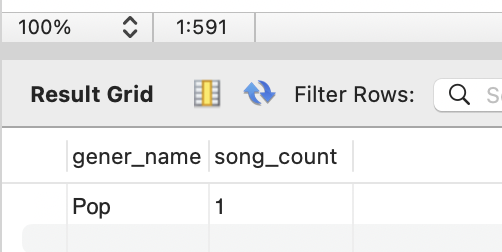
FROM Gener g

JOIN Song s ON g.gener\_id = s.gener\_id

GROUP BY g.gener\_name

ORDER BY song\_count DESC

LIMIT 1;



**Query 23:** Window Function - Calculate Cumulative Average Ratings for Songs by Artist **Description:** Calculate the cumulative average ratings for songs by each artist, ordered by creation date.

**SQL Statement:**

SELECT a.artist\_name, s.song\_name, r.rating\_value,

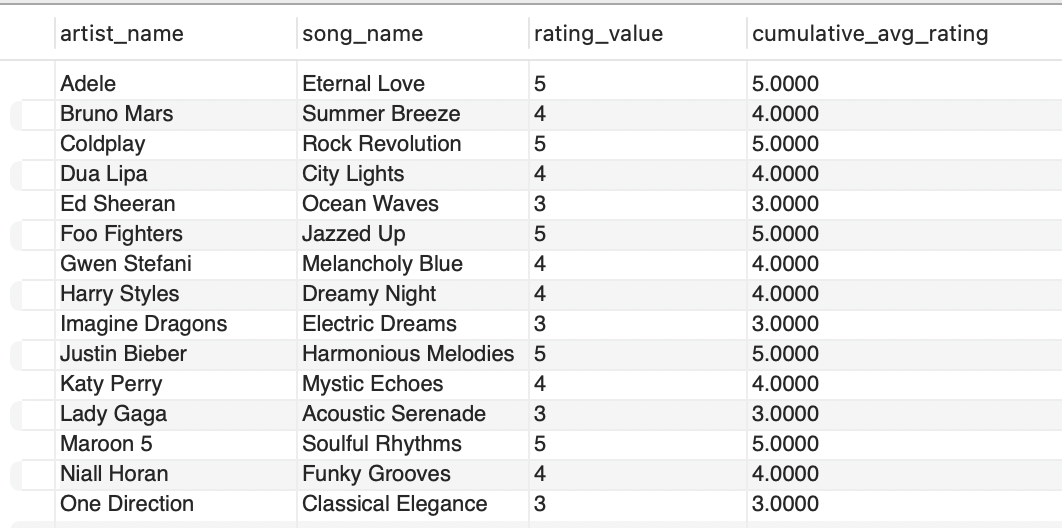
AVG(r.rating\_value) OVER (PARTITION BY a.artist\_name) AS cumulative\_avg\_rating

FROM Artist a

JOIN Song s ON a.artist\_id = s.artist\_id

JOIN Rating r ON s.song\_id = r.song\_id

ORDER BY a.artist\_name;

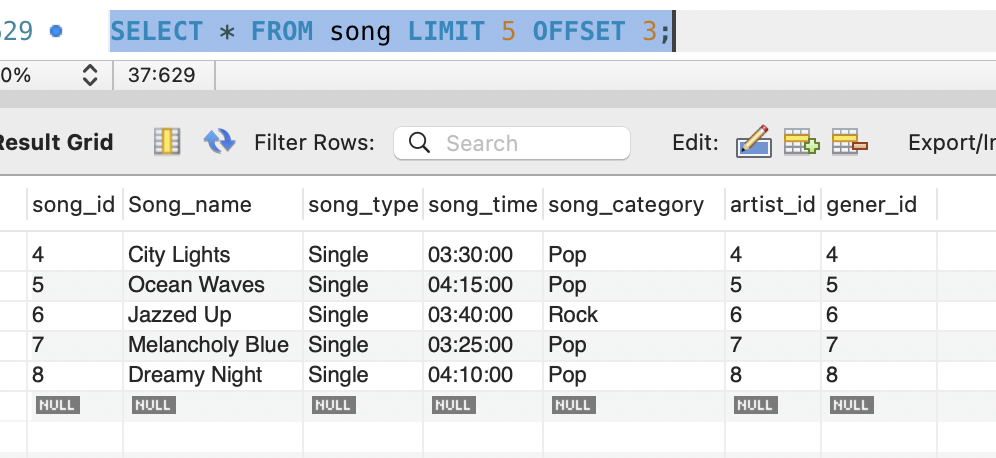


**Query 24:** Retrieve Top N Records with Offset

**Description:** Retrieve the top 5 records from a table, skipping the first 3.

**SQL Statement:**

SELECT \* FROM song LIMIT 5 OFFSET 3;



**Query 25:** Retrieve Artists with Songs in Multiple Languages

**Description:** Find artists who have songs in more than one language.

**SQL Statement:**

SELECT DISTINCT a.artist\_name

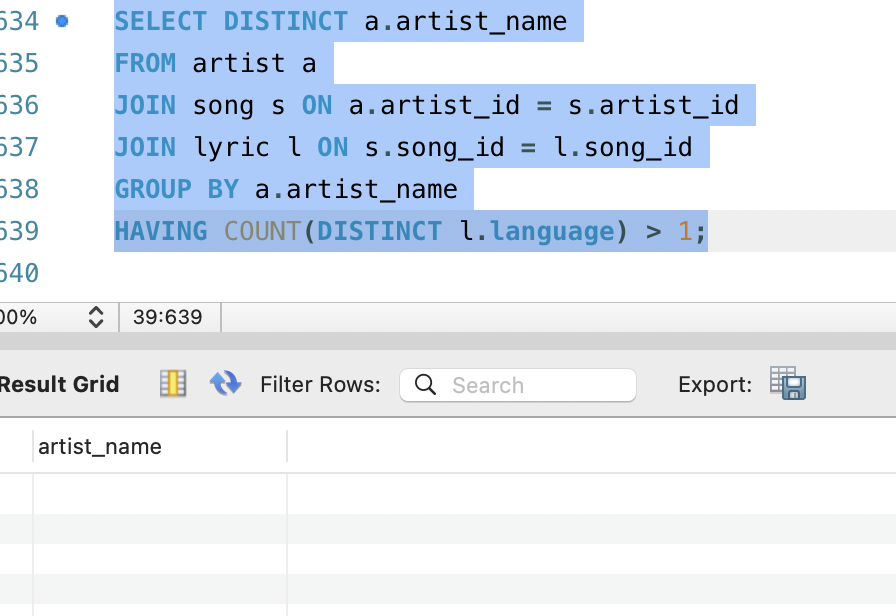
FROM artist a

JOIN song s ON a.artist\_id = s.artist\_id

JOIN lyric l ON s.song\_id = l.song\_id

GROUP BY a.artist\_name

HAVING COUNT(DISTINCT l.language) > 1;



Since in the given data we only have english language we got null value.

**Team G/Project number 25:**

1. Swathi Yadlapalli - A20545355
2. Sai Sivani Dukkipati - A20547981
3. Nallagangula Venkata Kishore Kumar Reddy - A20545723
4. Molli Venkata Naga Satya Dinesh Datta - A20546385
5. Sanjana Reddy- A20550437

**Team Contribution:**

| **Name** | **Contribution** |
| --- | --- |
| Swathi Yadlapalli | 20% |
| Sai Sivani Dukkipati | 20% |
| Kishore Kumar Reddy | 20% |
| Dinesh Datta Molli | 20% |
| Sanjana Reddy | 20% |