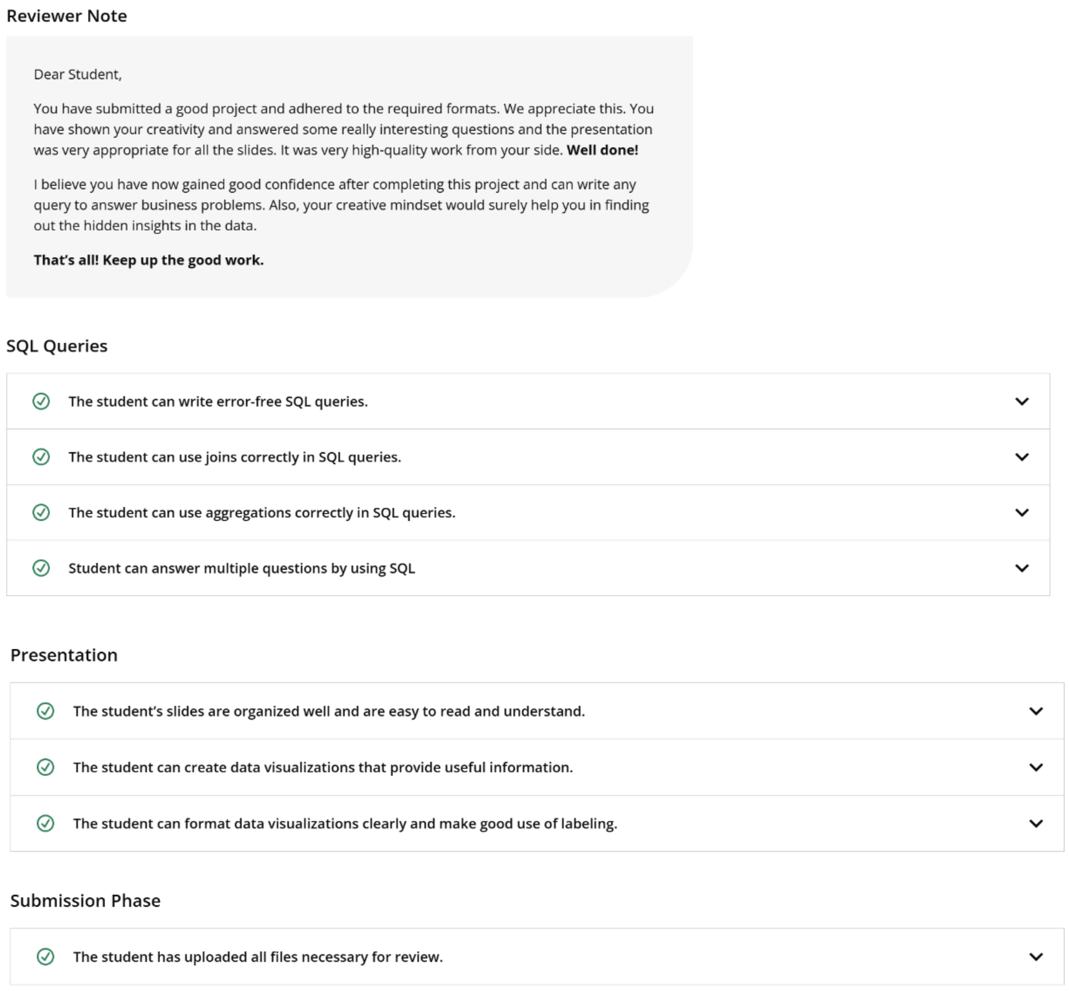
Reviewer Comments



Reviewer Note

Dear Student,

You have submitted a good project and adhered to the required formats. We appreciate this. You have shown your creativity and answered some really interesting questions and the presentation was very appropriate for all the slides. It was very high-quality work from your side. **Well done!**

I believe you have now gained good confidence after completing this project and can write any query to answer business problems. Also, your creative mindset would surely help you in finding out the hidden insights in the data.

That's all! Keep up the good work.

SQL Queries

- ✓ **The student can write error-free SQL queries.**
- ✓ **The student can use joins correctly in SQL queries.**
- ✓ **The student can use aggregations correctly in SQL queries.**
- ✓ **Student can answer multiple questions by using SQL**

Presentation

- ✓ **The student's slides are organized well and are easy to read and understand.**
- ✓ **The student can create data visualizations that provide useful information.**
- ✓ **The student can format data visualizations clearly and make good use of labeling.**

Submission Phase

- ✓ **The student has uploaded all files necessary for review.**

5 MANNIE BROWN queries for Udacity Project 2 Business Analytics Music Records.txt

```
1  /* Query 1 - query used for first insight /
2  
3  SELECT G.Name,
4         T.GenreId,
5         COUNT(T.GenreID) AS No_of_Tracks,
6         (COUNT(T.genreid) *T.UnitPrice) AS Genre_Earnings_Potential
7  FROM Genre G
8  JOIN Track T
9  ON T.GenreId = G.GenreId
10 GROUP BY T.GenreId
11 ORDER BY Genre_Earnings_Potential DESC
12 LIMIT 10;
```

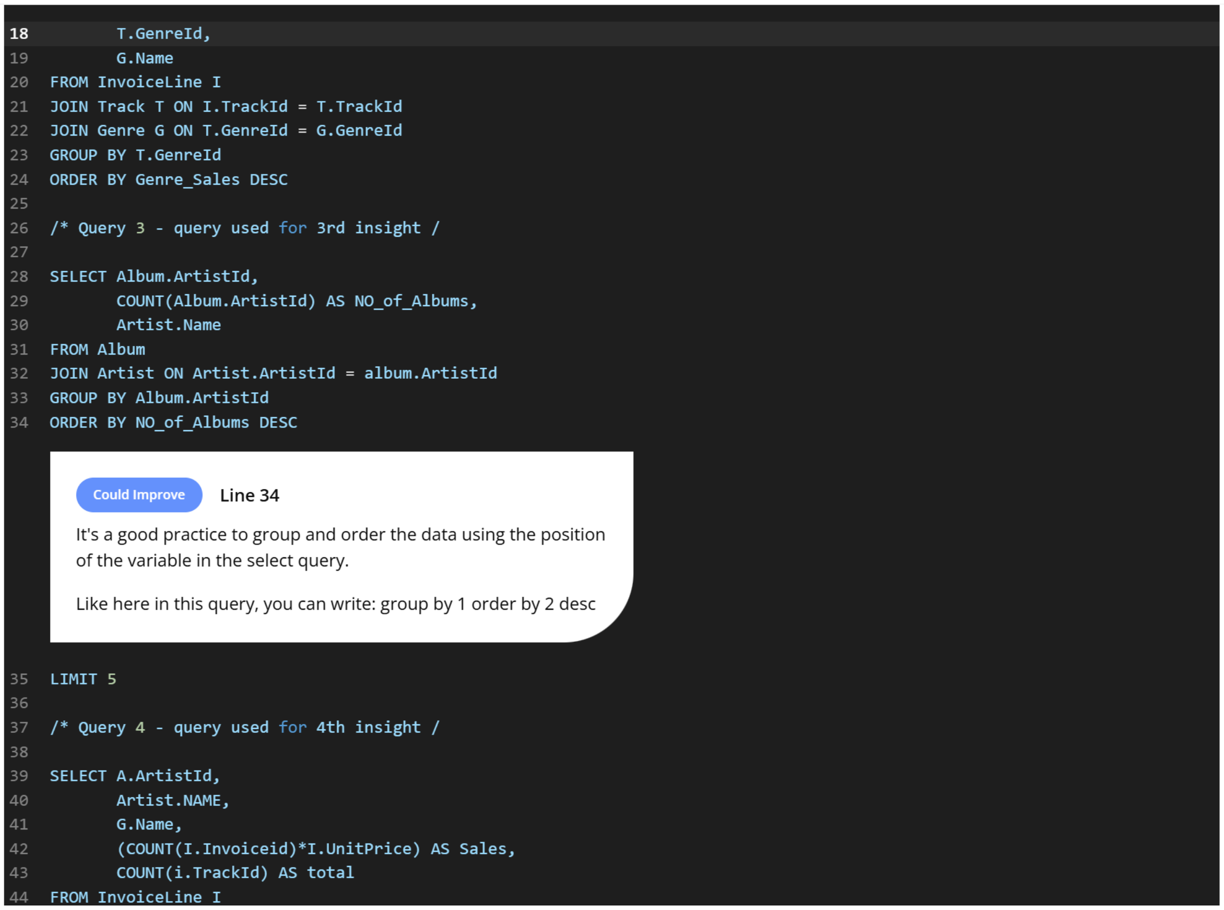
Great Line 12

It is always a good practice to limit the output for the top/bottom few entries as it helps in creating a neat chart. **Great Job!!!**

```
13 
14 
15 /* Query 2 - query used for 2nd insight /
16 
17 SELECT SUM(I.UnitPrice * I.Quantity) AS Genre_Sales,
```

Great Line 17

Awesome! You have correctly implemented the logic to compute the total spending.



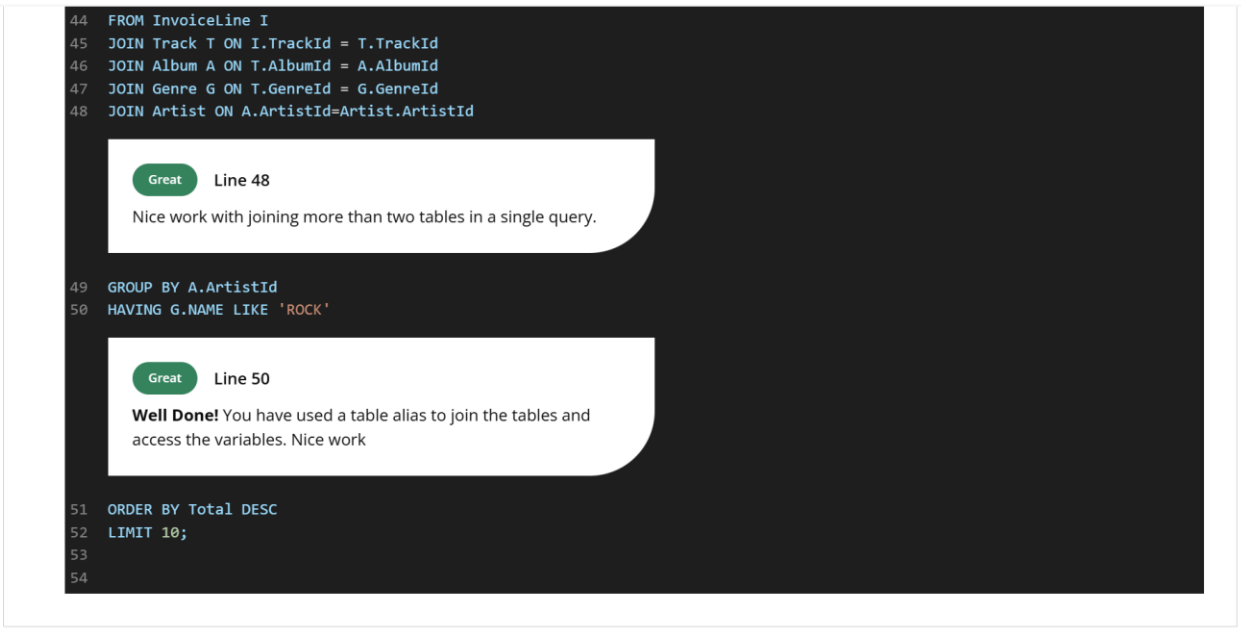
```
18         T.GenreId,
19         G.Name
20  FROM InvoiceLine I
21  JOIN Track T ON I.TrackId = T.TrackId
22  JOIN Genre G ON T.GenreId = G.GenreId
23  GROUP BY T.GenreId
24  ORDER BY Genre_Sales DESC
25
26  /* Query 3 - query used for 3rd insight /
27
28  SELECT Album.ArtistId,
29         COUNT(Album.ArtistId) AS NO_of_Albums,
30         Artist.Name
31  FROM Album
32  JOIN Artist ON Artist.ArtistId = album.ArtistId
33  GROUP BY Album.ArtistId
34  ORDER BY NO_of_Albums DESC
```

Could Improve Line 34

It's a good practice to group and order the data using the position of the variable in the select query.

Like here in this query, you can write: group by 1 order by 2 desc

```
35  LIMIT 5
36
37  /* Query 4 - query used for 4th insight /
38
39  SELECT A.ArtistId,
40         Artist.NAME,
41         G.Name,
42         (COUNT(I.Invoiceid)*I.UnitPrice) AS Sales,
43         COUNT(i.TrackId) AS total
44  FROM InvoiceLine I
```



```
44  FROM InvoiceLine I
45  JOIN Track T ON I.TrackId = T.TrackId
46  JOIN Album A ON T.AlbumId = A.AlbumId
47  JOIN Genre G ON T.GenreId = G.GenreId
48  JOIN Artist ON A.ArtistId=Artist.ArtistId
```

Great Line 48

Nice work with joining more than two tables in a single query.

```
49  GROUP BY A.ArtistId
50  HAVING G.NAME LIKE 'ROCK'
```

Great Line 50

Well Done! You have used a table alias to join the tables and access the variables. Nice work

```
51  ORDER BY Total DESC
52  LIMIT 10;
53
54
```