Module 4 Assignment Solution

A.Optimizing Music Database Queries for Enhanced Data Analysis

1. SELECT track.track_id, track.name AS track_name, genre.name AS genre_name
FROM track
INNER JOIN genre ON track.genre_id = genre.genre_id;

2. SELECT track.track_id, track.name AS track_name, genre.name AS genre_name
FROM track
LEFT JOIN genre ON track.genre_id = genre.genre_id;

3. SELECT track.track_id, track.name AS track_name, genre.name AS genre_name
FROM track
RIGHT JOIN genre ON track.genre_id = genre.genre_id;

4. SELECT track.track_id, track.name AS track_name, genre.genre_id, genre.name AS genre_name
FROM track
CROSS JOIN genre;

5. SELECT name FROM track WHERE milliseconds > 300000
UNION
SELECT name FROM track WHERE unit_price > 0.99;

6. SELECT name FROM track WHERE milliseconds > 300000
UNION ALL
SELECT name FROM track WHERE unit_price > 0.99;

7. SELECT track.track_id, track.name AS track_name, genre.name AS genre_name

```
FROM track
INNER JOIN genre ON track.genre_id = genre.genre_id
WHERE genre.name = 'Rock';

8. SELECT genre.name AS genre_name, COUNT(track.track_id) AS track_count
FROM genre
LEFT JOIN track ON genre.genre_id = track.genre_id
GROUP BY genre.name;

9. SELECT genre.genre_id, genre.name AS genre_name
FROM genre
LEFT JOIN track ON genre.genre_id = track.genre_id
WHERE track.track_id IS NULL;

10. SELECT track.track_id, track.name AS track_name, genre.genre_id, genre.name
AS genre_name
FROM track
CROSS JOIN genre
WHERE genre.genre_id NOT IN (SELECT DISTINCT genre_id FROM track);
```

B.Utilizing Advanced Data Analysis on Titanic Data

```
1. SELECT passenger_id, first_name, last_name, fare
FROM titanic
WHERE fare > (SELECT AVG(fare) FROM titanic);

2. SELECT passenger_id, first_name, last_name, pclass
FROM titanic
WHERE pclass = (SELECT pclass FROM titanic WHERE first_name = 'Julia' AND last_name =
```

'Patel');

3. SELECT passenger_id, first_name, last_name, embarked_town
FROM titanic
WHERE embarked_town = (SELECT embarked_town FROM titanic GROUP BY
embarked_town ORDER BY COUNT(*) DESC LIMIT 1);

4. SELECT passenger_id, first_name, last_name, age
FROM titanic
WHERE survived = 1 AND age < (SELECT AVG(age) FROM titanic);

5. SELECT passenger_id, first_name, last_name, fare
FROM titanic
WHERE fare IN (SELECT fare FROM titanic ORDER BY fare DESC LIMIT 10);

6. SELECT passenger_id, first_name, last_name, pclass
FROM titanic
WHERE pclass IN (
SELECT pclass
FROM titanic
GROUP BY pclass

HAVING AVG(survived) > (SELECT AVG(survived) FROM titanic)
);

7. SELECT passenger_id, first_name, last_name, deck, fare
FROM titanic
WHERE deck IN (
SELECT deck
FROM titanic
GROUP BY deck

```
ORDER BY AVG(fare)
LIMIT 1
);

8. SELECT passenger_id, first_name, last_name, pclass, fare
FROM titanic t1
WHERE fare > (SELECT AVG(fare) FROM titanic t2 WHERE t1.pclass = t2.pclass);

9. SELECT passenger_id, first_name, last_name, age, embarked_town
FROM titanic t1
WHERE age = (SELECT AVG(age) FROM titanic t2 WHERE t1.embarked_town =
t2.embarked_town);

10. SELECT passenger_id, first_name, last_name, deck_number
FROM titanic
WHERE deck_number = (SELECT deck_number FROM titanic GROUP BY deck_number
ORDER BY COUNT(*) DESC LIMIT 1);
```

C.Advanced Data Analysis on Titanic Dataset Using Window Functions

```
1. SELECT passenger_id, first_name, last_name, pclass,
LEAD(passenger_id) OVER (PARTITION BY pclass ORDER BY
passenger_id) AS next_passenger_id
FROM titanic;

2. SELECT passenger_id, first_name, last_name, pclass, fare,
LAG(fare) OVER (PARTITION BY pclass ORDER BY passenger_id) AS
previous_fare
FROM titanic;
```

3. SELECT passenger_id, first_name, last_name, pclass, fare,
RANK() OVER (PARTITION BY pclass ORDER BY fare DESC) AS fare_rank
FROM titanic;

4. SELECT passenger_id, first_name, last_name, pclass, age,
DENSE_RANK() OVER (PARTITION BY pclass ORDER BY age DESC) AS
age_dense_rank
FROM titanic;

5. SELECT passenger_id, first_name, last_name, embarked_town,
ROW_NUMBER() OVER (PARTITION BY embarked_town ORDER BY
passenger_id) AS row_num
FROM titanic;

6. SELECT passenger_id, first_name, last_name, pclass, sex,
LEAD(passenger_id) OVER (PARTITION BY pclass ORDER BY
passenger_id) AS next_female_passenger_id
FROM titanic
WHERE sex = 'female';

7. SELECT passenger_id, first_name, last_name, pclass, age, sex,
LAG(age) OVER (PARTITION BY pclass ORDER BY passenger_id) AS
previous_male_age
FROM titanic
WHERE sex = 'male';

8. SELECT passenger_id, first_name, last_name, pclass, fare, sex,
RANK() OVER (PARTITION BY pclass ORDER BY fare DESC) AS
female_fare_rank
FROM titanic
WHERE sex = 'female';

9. SELECT passenger_id, first_name, last_name, pclass, age, survived,
DENSE_RANK() OVER (PARTITION BY pclass ORDER BY age DESC) AS
survivor_age_dense_rank
FROM titanic
WHERE survived = 1;

10. SELECT passenger_id, first_name, last_name, embarked_town,

ROW_NUMBER() OVER (PARTITION BY embarked_town ORDER BY
passenger_id) AS row_num
FROM titanic
WHERE embarked_town = 'Southampton';