NetEase Cloud Music User Database

University of Connecticut
School of Business

OPIM5272 - Data Management and Business Process Modeling

Dr. Ugo Etudo

Spring 2020

SQL101

Simeng Deng

Lei Cao

Zhuo Chen

Contents

Introduction 4

Business Rules 4

User Requirements 5

EERD with Detailed Definitions 8

1. Definition of Entities 8
2. Definition of Attributes 9
3. Definition of Relationships 14

Relations At Least in the 3rd Normal Form 18

Tables and Records 18

Sample SQL Queries and Results 23

1. Our Members' Basic Information 23
2. Our Independent Artists’ Basic Information 24
3. Sex Composition 24
4. Each Member's Membership Duration 24
5. Listening History of Our Members 25
6. Where the User Accessed the Songs 25
7. The Performance of the Recommendation List 27
8. Information of Premium Songs 27
9. Songs’ Genres Composition 28

10. Information of Our Songs Including Duration of the Copyright 28

11. Members' Favorite Songs 29

12. Listening Duration 29

13. Count the Number of Times A Single Song is Liked by Users 31

14. Sum Listening Percentage of Each Song 32

15. Monthly Royalties Paid to Artists 32

PL/SQL and Results 33

1. Use Percentage to See if the User Likes the Song 33

2. Judge Membership Type using the MEMBERSHIP Function 34

3. If A User Listens to A Song Because of Our Recommendation List 35

4. Our Users' Genders and Ages 36

5. Users' Attitudes to the Songs 38

Additional Features Implemented 39

1. Print if the User Listened to A Song Because of the Recommendation List 39

2. Print Information of the Membership Payment 40

Introduction

NetEase Cloud Music is a widely used music app that provides Chinese music streaming services. It operates under a freemium business model where general users get basic services for free while paid members can enjoy some enhanced features, such as the access to premium music. Accumulated 800 million people are using this app so it holds a large bunch of user data. Faced with competition from Tencent's QQ Music, it is important for NetEase Cloud Music to build a user database to track user behaviors and analyze the key factors which contribute to improving the Stickiness and Freemium Conversion Rate.

Firstly, higher Stickiness represents users are more willing to engage with the APP, in other words, APP could attract more users. On the one hand, a larger number of users could further increase the platform’s competitiveness so that NetEase Cloud Music could contend for more copyrights of music. Besides, an active platform could attract more independent artists who publish their original songs here and some of them are rising stars that could bring unexpected popularity at a low cost. Finally, higher Freemium Conversion Rate enables the platform to gain more profit and develop sustainably.

Business Rules

1. There are two types of accounts that users can create on NetEase Cloud Music platform, one of which is for artists, while the other is for normal listeners.
2. Both artists and normal listeners are able to stream and download normal music for free, but for certain music the normal listeners have to pay for membership to listen to or download. Whereas artists can get access to all music on the platform without membership.

3. The artists include the signed and the unsigned. Both of them are able to upload and issue their music on the platform.
4. NetEase Cloud Music has to pay royalties to the artists based on the streams for their music issued on the platform. Royalties paid to the signed artists are higher than those to the independent ones.
5. The users can create playlists to collect their favorite music in various classifications.
6. According to users' listening preferences, NetEase Cloud Music's intelligent system generates daily recommendations of music to individuals.
7. Each time a user listened to one song, the time the user started, the time the user stopped and the point when the user stopped would be recorded. If the user played one song round, each round would be seen as one record.
8. The users may listen to a song from different sources, such as their playlists, the systematic recommendations, as well as the sharing from other users.
9. The users can also deliver their reviews to a particular song with a "Like" or "Dislike" trigger.

User Requirements

The database stored detailed information of NetEase Cloud Music users and issued music which could be used to analyze users' features, users' behaviors, songs' performance etc. Tables for analysis can be generated by SELECT statements and PL/SQL procedures.

- User's Feature

1) Member's information: gather all detailed information of member users.

2) Independent artist's information: gather all detailed information of independent artists.

3) Sex composition: count the number of male users and female users.

- Users' Behaviors

1) The cumulative duration of membership: count the cumulative duration of membership for each normal user.

2) Listening history of the members: gather all listening history of members.

3) Sources: collect the sources where the user accessed the song each time when the song was being heard.

4) The performance of recommendation list: collect songs which are recommended to users through recommendation lists and being favored by the users.

- Music

1) Collection of Premium songs: gather the information of premium songs.

2) Music genre composition: count the number of different music genres.

3) Copyright duration: calculate the copyright duration for each song.

4) Members' favorite songs: gather the listening records of songs favored by members.

5) Listening duration: calculate the duration for each listening record.

6) Songs liked: count the number of times a single song liked by users.

7) Songs' performance: calculate the average listening duration percentage[1] and played times of each song.

[1] Average listening duration percentage = Average percentage of listening duration divided by the length of the song.

8) Monthly royalties paid to artists: calculate the monthly royalties[2] paid to artists.

- PL/ SQL

1) A procedure named USERPREFQUANT based on the percentage, which is derived from length of song and the time user stops when listening to a song, to judge whether the user likes the song. When the percentage exceeded 0.5, it determined that the user liked the song.
2) A function named MEMBERSHIP used to return the membership duration for each purchase of the user. The result can be monthly, half-year, and whole-year.
3) A function named RECO that can show if the user listened to the song because of our recommendation list.
4) A function named GENDERAGE used to know our users' gender and age. The result can be a boy, a girl, a gentleman, and a lady.
5) A procedure to tell the users' attitude (Like or Dislike) to songs.
6) Combine our RECO function and selection from table to see if the user listened to the song because of our recommendation list or not.
7) Combine our MEMBERSHIP function and selection from table to generate a running account about membership purchases.

[2] Monthly Royalties = Monthly valid times of streams (listening duration >30s) * corresponding royalty

EERD with Detailed Definitions

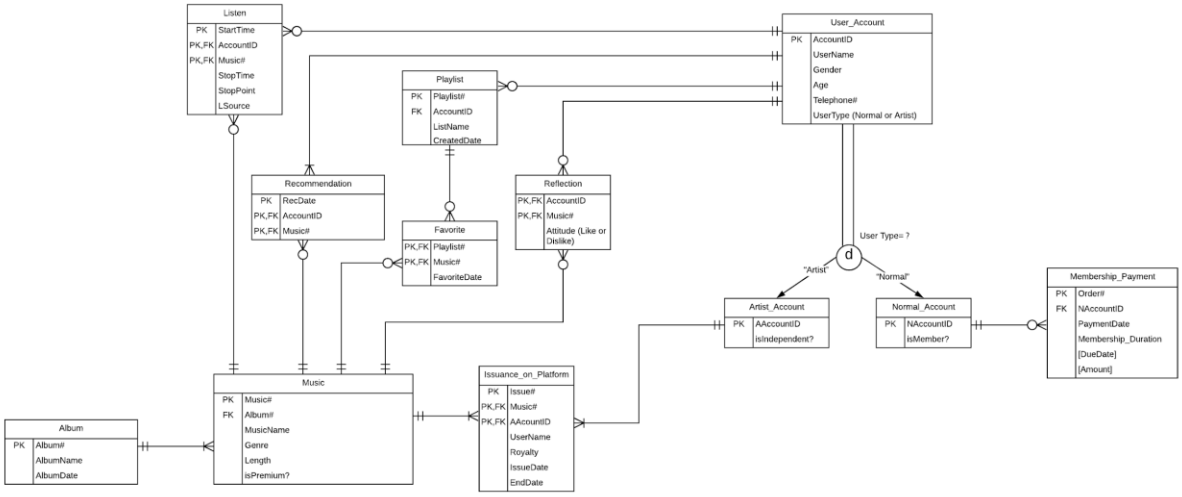

Figure 1 EERD

1. Definition of Entities

Entity	Definition
User_Account	Records users' general information for registration on NetEase Cloud Music. The supertype of the Artist_Account and Normal_Account entities.
Artist_Account	One subtype of the User_Account, recording artists' general information and whether the artists are independent of record companies.
Normal_Account	The other subtype of the User_Account, recording normal users' general information and whether the users are members.
Membership_Payment	Keeps track of normal users' payment for membership.

Issuance_on_Platform	An associative entity between the Music and the Artist_Account to record the issue information of the music once an artist issues music on the platform.
Music	An archive gathering all music that is available on NetEase Cloud Music.
Album	Records the album information of the corresponding music.
Recommendation	Captures the specific music the NetEase Cloud Music system recommended daily to individuals and the recommended date.
Listen	Keeps track of the listening behaviors of the users.
Favorite	An associative entity between the Playlist and the Music triggered by users when they are adding music to their personal playlists.
Playlist	Records the information of playlists created by users to collect their favorite music.
Reflection	Captures the user's attitude towards a particular song (whether the user likes or dislikes it).

2. Definition of Attributes

- User_Account

Attribute	Definition
AccountID	Each user account is uniquely identified by AccountID.
UserName	The name of the account set by the user.
Gender	The gender set by the account user.
Age	The age set by the account user.
Telephone	The telephone number used to apply for this account.
UserType	Identification of whether the user is a normal user or an artist who would publish songs. User Type can be Artist or Normal.

- Artist_Account

Attribute	Definition
AAccountID	The account number of an artist user.
isIndependent?	A binary attribute recording whether the artist has signed a company or independent singer. “Y” is for independent artists, while “N” is for signed artists.

- Normal_Account

Attribute	Definition
NAccountID	The account number of a normal user.
isMember?	A binary attribute recording whether the normal user account is in the period of membership. “Y” means in membership period, while “N” means not in membership period.

- Membership_Payment

Attribute	Definition
Order#	Each payment is uniquely identified by the order number.
NAccountID	The accountID of the normal user who paid for the membership.
PaymentDate	The Date that this payment occurred.
Membership_Duration	How long of the membership the user paid. The Membership Duration can be either 1 month, half year or one year. The unit of Membership Duration is month.

DueDate	Duedate is calculated using PaymentDate and Membership_Duration. The formula is that DueDate equals to PaymentDate adds Membership_Duration.
Amount	The payment amount is calculated using Membership_Duration. The formula is Membership_Duration *Monthly Rate* Discount Rate. The Monthly Rate is $10. If the user paid for one month, the discount rate equals 1; If the user paid for half a year, the discount rate equals 0.8; If the user paid for one year, the discount rate equals 0.7.

- Issuance_on_Platform

Attribute	Definition
Issue#	An issue number is part of the identifier of the issuance of a song on the platform.
Music#	The music number of a song that is issued on the platform. It is also part of the identifier of the Issuance_on_Platform.
AAccountID	The account number of the artist who issued the song. It is also part of the identifier of the Issuance_on_Platform.
UserName	The account name of the artist who issued the song.
Royalty	The royalty fee paid to the artist based on the streams of the song issued.
IssueDate	The date when the song was issued on the platform.
EndDate	The date when the song should be removed as the copyright is due.

- Music

Attribute	Definition

Music#	A number uniquely identifying a song on the platform.
Album#	The album number of an album a particular song belongs to.
MusicName	The name of a song.
Genre	The genre that a song is classified as.
Length	The length of the song in seconds.
isPremium?	A binary attribute recording whether a song is only accessible to members. “Y” represents that a song can only be streamed and downloaded by members, while “N” represents that a song can be streamed and downloaded by all users for free.

- Album

Attribute	Definition
Album#	A number uniquely identifying an album.
AlbumName	The name of an album.
AlbumDate	The date of the issue of an album.

- Recommendation

Attribute	Definition
RecDate	Date of a recommendation generated. It is part of the identifier of the Recommendation.
AccountID	The account number of the user who received this recommendation. It is also part of the identifier of the Recommendation.
Music#	The music number of the song that is recommended. It is also part of the identifier of the Recommendation.

- Listen

Attribute	Definition
StartTime	The Time Stamp that the user started listening to one song. It is part of the identifier of the Listen.
AccountID	The account number of the user who listened to this song at this time. It is also part of the identifier of the Listen.
Music#	The music number of the song that the user listened to at this time. It is also part of the identifier of the Listen.
StopTime	The Time Stamp that the user stopped listening to the song.
StopPoint	The seconds format of the time point that the user stopped listening to this song.
LSource	How the user accessed this song. The LSource can be either Recommendation List, Search, Playlist and Other Recommendation.

- Favorite

Attribute	Definition
Playlist#	The playlist number of the playlist that a user added his or her favorite music in. It is part of the identifier of the Favorite.
Music#	The music number of the song that is added by the user as his or her favorite. It is also part of the identifier of the Favorite.
FavoriteDate	The date when the user added the music as a favorite.

- Playlist

Attribute	Definition

Playlist#	A number uniquely identifying a playlist created by a user on the platform.
AccountID	The account number of the user who created the playlist.
ListName	The name that the user set to the created playlist.
CreatedDate	The date when the user created the playlist.

- Reflection

Attribute	Definition
AccountID	The account number of a user who delivered his or her attitude to a song. It is part of the identifier of the Reflection.
Music#	The music number of the song that the user reviewed. It is also part of the identifier of the Reflection.
Attitude	A binary attribute indicating whether the user liked or disliked the song. “Like” represents that the user liked the song, while “Dislike” represents that the user did not like the song.

3. Definition of Relationships

- User_Account & Artist_Account / Normal_Account

Supertype	Disjoint Total Specialization	Subtype
User_Account	User Type = Artist	Artist_Account
	User Type = Normal	Normal_Account
The User_Account has two disjoint subtypes, one of which is the Normal_Account and the other is the Artist_Account.		

- Normal_Account & Membership_Payment

Normal_Account	Mandatory 1	Optional n	Membership_Payment
A normal user may or may not pay for membership. One user may pay for membership more than once. And each payment of membership must be triggered only by one user.			

- Issuance_on_Platform & Artist_Account

Artist_Account	Mandatory 1	Mandatory n	Issuance_on_Platform
An artist must issue a work through his or her account, otherwise the artist account would not be activated. An artist may issue more than once, and an issuance is delivered by one artist.			

- Issuance_on_Platform & Music

Music	Mandatory 1	Mandatory n	Issuance_on_Platform
Each issuance must be related to a song, while a song may have more than one issuance if it is composed by more than one artist.			

- Music & Album

Album	Mandatory 1	Mandatory n	Music
Each song must belong to an album, and each album has at least one song.			

- Listen & Music

Music	Mandatory 1	Optional n	Listen
Each time of listening must be triggered by only one song, but a song may or may not be listened to. And a song can be listened to many times.			

- Listen & User_Account

User_Account	Mandatory 1	Optional n	Listen
A user may or may not listen to music, and he or she would listen to music many times. Meanwhile each time of listening must be related to only a user.			

- Recommendation & Music

Music	Mandatory 1	Optional n	Recommendation
One recommendation must be aimed at one song, whereas a song may or may not be recommended. And a song can be recommended more than once.			

- Recommendation & User_Account

User_Account	Mandatory 1	Optional n	Recommendation
A user will receive at least one daily recommendation of music, but each record of recommendation must be in relation to only one user.			

- Favorite & Music

Music	Mandatory 1	Optional n	Favorite
Every record of a favorite must be linked to only one song, while a song may or may not be selected as a favorite. Plus, one song can be selected as a favorite many time.			

- Favorite & Playlist

Playlist	Mandatory 1	Optional n	Favorite
There may or may not be a song collected into a playlist, and a playlist can have more than one song. A record of a favorite must be triggered by only one playlist.			

- Playlist & User_Account

User_Account	Mandatory 1	Optional n	Playlist
A user may or may not create a playlist to collect music, and the user can create many playlists for distinct classifications. Each playlist must belong to only one user.			

- Reflection & Music

Music	Mandatory 1	Optional n	Reflection
A song may or may not be reviewed by users through expressing their attitudes, and a song can have many reviews from the users. But each time of reflection must be linked to only one song.			

- Reflection & User_Account

User_Account	Mandatory 1	Optional n	Reflection
A user may or may not review a song, and he or she can review many songs. But each record of reflection must come from only one user.			

Relations At Least in the 3rd Normal Form

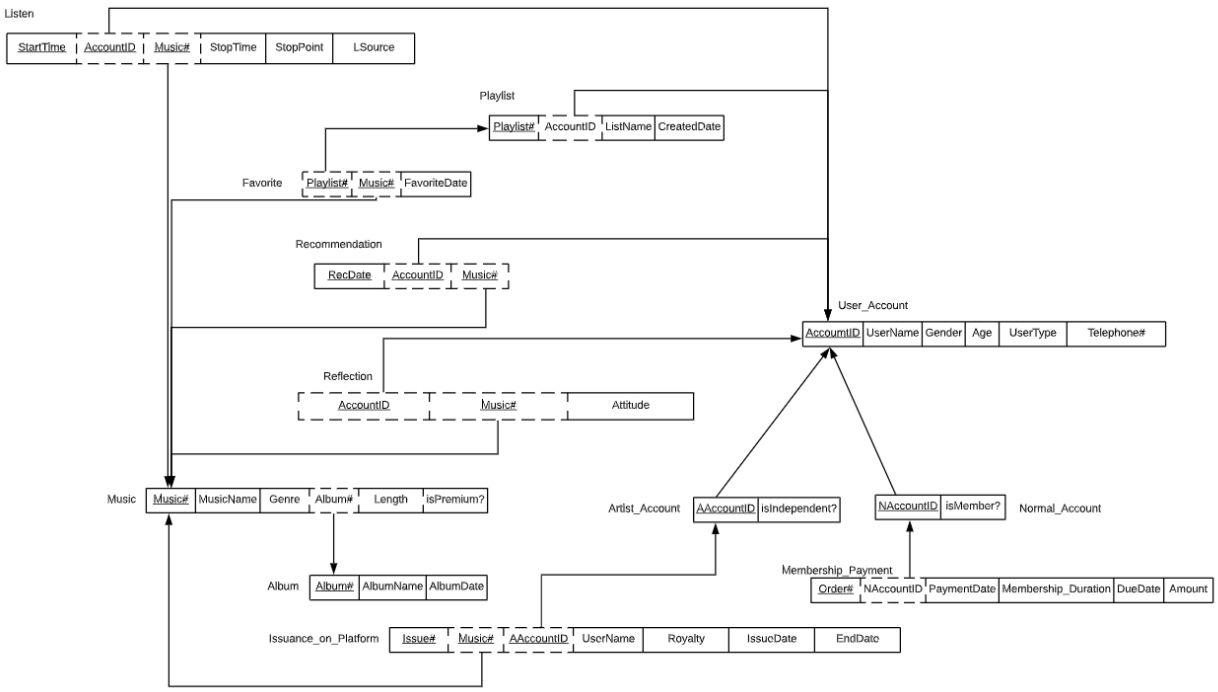

Figure 2 Relations

Tables and Records

- User_Account

	ACCOUNTID	USERNAME	GENDER	AGE	USERTYPE	TELEPHONE_NUMBER
1	12	Feng Wang	M	48	Artist	13945205106
2	11	Chunhua Xie	F	25	Artist	17876543946
3	10	Hui Tian	M	14	Normal	16956464832
4	9	Rosalie	F	19	Normal	2026543213
5	8	Simon	M	30	Normal	8325464231
6	7	Xuesi	F	22	Normal	13551613513
7	6	Lauv	M	25	Artist	8608970748
8	5	Bu Cai	F	(null)	Artist	13910959291
9	4	Avril Lavigne	F	35	Artist	8603241000
10	3	Cassie	F	27	Normal	15365465432
11	2	Ray	M	(null)	Normal	17068976546
12	1	Adam	M	16	Normal	13668465423

- Normal_Account

	NACCOUNTID	ISMEMBER
1	10	N
2	9	N
3	8	Y
4	7	N
5	3	Y
6	2	Y
7	1	Y

- Artist_Account

	AACCO...	ISINDEPENDENT
1	12	N
2	11	Y
3	6	N
4	5	Y
5	4	N

- Music

	MUSIC_NU...	MUSICNAME	GENRE	ALBUM_NUMBER	SONGLENGTH	ISPREMIUM
1	10	Second-Hand Soul	Rock	7	239	N
2	9	Not That Easy	Rock	7	242	Y
3	8	11887	Pop	6	272	N
4	7	Okay	Pop	5	205	Y
5	6	You Are	Pop	5	229	Y
6	5	Not a Lover	R&B	4	231	N
7	4	I Fell In Love With The Devil	Pop	3	255	Y
8	3	Head Above Water	Pop Rock	3	220	Y
9	2	Paris in the Rain	Pop	2	204	N
10	1	I Like Me Better	Pop	1	197	N

- Album

	ALBUM_NUMBER	ALBUMNAME	ALBUMDATE
1	7	2020	04/20/2020
2	6	Dim Sum	06/12/2018
3	5	Silent Rebellion	06/06/2019
4	4	Not a Lover	07/01/2019
5	3	Head Above Water	02/14/2019
6	2	Paris in the Rain	11/16/2017
7	1	I Like Me Better	05/18/2017

- Playlist

	PLAYLIST_NUMBER	ACCOUNTID	LISTNAME	CREATEDDATE
1	4	6	INSPIRING	04/30/2020
2	3	2	CALM	02/14/2020
3	7	7	COOL	01/20/2020
4	2	3	SAD	12/21/2019
5	8	8	SADNESS	10/19/2019
6	1	1	HAPPY	10/04/2019
7	9	11	ENERGETIC	09/30/2019
8	6	10	SORROW	07/15/2019
9	10	3	ROCK	06/02/2019
10	5	7	BEAUTIFUL	06/10/2018

- Favorite

	PLAYLIST_NUMBER	MUSIC_NUMBER	FAVORITEDATE
1	4	3	04/30/2020
2	10	9	04/30/2020
3	9	9	04/30/2020
4	7	10	04/20/2020
5	3	2	02/14/2020
6	2	4	12/31/2019
7	8	5	12/24/2019
8	2	5	12/21/2019
9	1	1	10/04/2019
10	6	5	08/25/2019
11	5	8	05/06/2019

- Issuance_on_Platform

	ISSUE_NUMBER	MUSIC_NUMBER	AACCOUNTID	USERNAME	ROYALTY	ISSUEDATE	ENDDATE
1	9	9	12	Feng Wang	0.005	04/20/2020	04/20/2050
2	10	10	12	Feng Wang	0.005	04/20/2020	04/20/2050
3	5	5	5	Bu Cai	0.001	07/01/2019	07/01/2049
4	8	8	11	Chunhua Xie	0.001	06/12/2019	06/12/2049
5	6	6	11	Chunhua Xie	0.001	06/06/2019	06/06/2049
6	7	7	11	Chunhua Xie	0.001	06/06/2019	06/06/2049
7	4	4	4	Avril Lavigne	0.006	02/14/2019	02/14/2049
8	3	3	4	Avril Lavigne	0.006	02/14/2019	02/14/2049
9	2	2	6	Lauv	0.003	11/16/2017	11/16/2047
10	1	1	6	Lauv	0.003	05/18/2017	05/18/2047

- Membership_Payment

	ORDER_NUMBER	NACCOUNTID	PAYMENTDATE	MEMBERSHIP_DURATION	DUEDATE	AMOUNT
1	8	8	05/08/2020	1	06/07/2020	10
2	7	8	04/07/2020	1	05/07/2020	10
3	6	1	05/02/2020	12	04/27/2021	84
4	5	3	04/03/2020	12	03/29/2021	84
5	4	2	12/07/2019	6	06/04/2020	48
6	3	2	11/06/2019	1	12/06/2019	10
7	2	3	10/02/2019	6	03/30/2020	48
8	1	2	10/05/2019	1	11/04/2019	10

- Reflection

	ACCOUNTID	MUSIC_NUMBER	ATTITUDE
1	12	3	Dislike
2	10	5	Like
3	9	5	Dislike
4	8	4	Like
5	8	6	Dislike
6	7	5	Like
7	7	1	Like
8	7	8	Like
9	6	2	Like
10	6	4	Dislike
11	3	5	Like
12	3	9	Like
13	2	1	Dislike
14	1	4	Like
15	1	3	Like

- Recommendation

	RECDATE	ACCOUNTID	MUSIC_NUMBER
1	08/31/2019	11	4
2	11/01/2020	10	10
3	11/01/2020	9	1
4	01/11/2020	9	1
5	02/29/2020	8	6
6	03/18/2020	7	5
7	07/15/2019	7	1
8	01/11/2020	6	6
9	03/07/2020	6	1
10	04/09/2020	5	4
11	12/19/2019	4	5
12	02/28/2020	3	9
13	12/31/2019	3	5
14	08/31/2019	3	1
15	03/18/2020	2	10
16	01/28/2020	2	3
17	10/20/2019	1	4
18	10/04/2020	1	1
19	10/04/2019	1	2
20	03/18/2020	1	5

- Listen

	STARTTIME	ACCOUNTID	MUSIC_NUMBER	STOPTIME	STOPPOINT	LSOURCE
1	05/02/2020 07:20:59	8	9	05/02/2020 07:22:01	62	Search
2	04/30/2020 15:28:40	6	3	04/30/2020 15:31:30	170	Search
3	04/30/2020 15:25:00	6	3	04/30/2020 15:28:40	220	Search
4	04/30/2020 00:00:21	3	9	04/30/2020 00:04:23	242	Playlist
5	04/25/2020 03:00:04	11	9	04/25/2020 03:04:06	242	Playlist
6	04/20/2020 17:50:11	3	9	04/20/2020 17:54:13	242	Recommendation List
7	04/20/2020 02:45:02	7	10	04/20/2020 02:49:01	239	Playlist
8	04/15/2020 01:01:58	11	4	04/15/2020 01:06:13	291	Other Recommendation
9	04/04/2020 22:21:51	5	6	04/04/2020 22:25:40	229	Search
10	04/04/2020 15:11:20	7	1	04/04/2020 15:14:37	163	Search
11	04/04/2020 11:00:05	3	5	04/04/2020 11:03:56	231	Search
12	03/29/2020 22:48:47	1	1	03/29/2020 22:52:04	163	Playlist
13	03/18/2020 17:54:20	9	2	03/18/2020 17:56:31	131	Search
14	03/18/2020 00:21:11	7	5	03/18/2020 00:25:02	231	Recommendation List
15	02/29/2020 07:21:08	8	6	02/29/2020 07:22:06	58	Recommendation List

16	02/14/2020 14:03:40	2	2	02/14/2020 14:07:55	291	0ther Recommendation
17	02/14/2020 14:00:00	2	3	02/14/2020 14:03:40	220	0ther Recommendation
18	01/20/2020 14:49:11	8	5	01/20/2020 14:53:02	205	Playlist
19	01/19/2020 13:19:20	12	7	01/19/2020 13:22:45	205	0ther Recommendation
20	12/31/2019 10:36:37	3	5	12/31/2019 10:40:28	231	Recommendation List
21	12/31/2019 10:36:37	3	1	12/31/2019 10:38:20	103	Search
22	12/31/2019 10:33:20	3	1	12/31/2019 10:36:37	197	Search
23	12/31/2019 10:30:00	3	4	12/31/2019 10:33:20	200	Search
24	10/04/2019 19:04:17	1	1	10/04/2019 19:07:00	163	Recommendation List
25	10/04/2019 19:01:00	1	1	10/04/2019 19:04:17	163	Recommendation List
26	10/04/2019 19:00:00	1	2	10/04/2019 19:01:00	60	Recommendation List
27	08/31/2019 22:46:36	11	4	08/31/2019 22:50:51	291	Recommendation List
28	08/31/2019 22:42:21	11	4	08/31/2019 22:46:36	291	Recommendation List
29	08/31/2019 22:38:06	11	4	08/31/2019 22:42:21	291	Recommendation List
30	08/31/2019 20:30:59	10	5	08/31/2019 20:34:50	231	Playlist

31	08/31/2019 20:27:08	10	5	08/31/2019 20:30:59	231	Playlist
32	08/31/2019 20:23:17	10	5	08/31/2019 20:27:08	231	Playlist
33	08/31/2019 20:19:26	10	5	08/31/2019 20:23:17	231	Playlist
34	08/31/2019 20:15:35	10	5	08/31/2019 20:19:26	231	Playlist
35	07/19/2019 09:10:50	12	3	07/19/2019 09:11:31	41	Search
36	07/15/2019 03:32:27	7	1	07/15/2019 03:34:44	137	Recommendation List
37	07/15/2019 03:29:10	7	1	07/15/2019 03:32:27	163	Recommendation List
38	07/15/2019 03:25:53	7	1	07/15/2019 03:29:10	163	Recommendation List
39	07/15/2019 03:22:36	7	1	07/15/2019 03:25:53	163	Recommendation List
40	07/15/2019 03:19:19	7	1	07/15/2019 03:22:36	163	Recommendation List
41	07/02/2019 02:16:08	9	5	07/02/2019 02:16:33	25	0ther Recommendation
42	06/08/2019 16:20:20	8	4	06/08/2019 16:22:34	134	0ther Recommendation
43	05/08/2019 01:19:49	7	8	05/08/2019 01:24:21	272	Playlist
44	05/08/2019 01:15:17	7	8	05/08/2019 01:19:49	272	Playlist
45	05/08/2019 01:10:45	7	8	05/18/2019 01:15:17	272	Playlist

Sample SQL Queries and Results

1. Our Members' Basic Information

```
SELECT *
FROM User_Account
JOIN Normal_Account ON User_Account.ACCOUNTID = Normal_Account.NACCOUNTID
WHERE Normal_Account.ISMEMBER = 'Y';
```

	ACCOUNTID	USERNAME	GENDER	AGE	USERTYPE	TELEPHONE_NUMBER	NACCOUNTID	ISMEMBER
1	1	Adam	M	16	Normal	13668465423	1	Y
2	2	Ray	M	(null)	Normal	17068976546	2	Y
3	3	Cassie	F	27	Normal	15365465432	3	Y
4	8	Simon	M	30	Normal	8325464231	8	Y

2. Our Independent Artists' Basic Information

```
SELECT *
FROM User_Account
JOIN Artist_Account on User_Account.ACCOUNTID = Artist_Account.AACCOUNTID
WHERE Artist_Account.ISINDEPENDENT = 'Y';
```

	ACCOUNTID	USERNAME	GENDER	AGE	USERTYPE	TELEPHONE_NUMBER	AACCOUNTID	ISINDEPENDENT
1	5	Bu Cai	F	(null)	Artist	13910959291	5	Y
2	11	Chunhua Xie	F	25	Artist	17876543946	11	Y

3. Sex Composition

```
SELECT GENDER, COUNT(GENDER) as GENDERTOTAL
FROM User_Account
GROUP BY GENDER;
```

	GENDER	GENDERTOTAL
1	M	6
2	F	6

4. Each Member's Membership Duration

```
SELECT NACCOUNTID, SUM(MEMBERSHIP_DURATION) as TotalMembershipDuration
FROM MEMBERSHIP_PAYMENT
GROUP BY MEMBERSHIP_PAYMENT.NACCOUNTID;
```

	NACCOUNTID	TOTALMEMBERSHIPDURATION
1	1	12
2	2	8
3	8	2
4	3	18

5. Listening History of Our Members

```
SELECT q1.ACCOUNTID, MUSIC_NUMBER, STARTTIME, STOPTIME, STOPPOINT,
LSOURCE
FROM LISTEN q1
INNER JOIN (SELECT ACCOUNTID
FROM User_Account
JOIN Normal_Account ON User_Account.ACCOUNTID = Normal_Account.NACCOUNTID
WHERE Normal_Account.ISMEMBER = 'Y') q2 on q1.ACCOUNTID = q2.ACCOUNTID;
```

	ACCOUNTID	MUSIC_NUMBER	STARTTIME	STOPTIME	STOPPOINT	LSOURCE
1	1	2	10/04/2019 19:00:00	10/04/2019 19:01:00	60	Recommendation List
2	1	1	10/04/2019 19:01:00	10/04/2019 19:04:17	163	Recommendation List
3	1	1	10/04/2019 19:04:17	10/04/2019 19:07:00	163	Recommendation List
4	3	4	12/31/2019 10:30:00	12/31/2019 10:33:20	200	Search
5	3	1	12/31/2019 10:33:20	12/31/2019 10:36:37	197	Search
6	3	1	12/31/2019 10:36:37	12/31/2019 10:38:20	103	Search
7	3	5	12/31/2019 10:36:37	12/31/2019 10:40:28	231	Recommendation List
8	2	3	02/14/2020 14:00:00	02/14/2020 14:03:40	220	Other Recommendation
9	2	2	02/14/2020 14:03:40	02/14/2020 14:07:55	291	Other Recommendation
10	8	4	06/08/2019 16:20:20	06/08/2019 16:22:34	134	Other Recommendation
11	8	5	01/20/2020 14:49:11	01/20/2020 14:53:02	205	Playlist
12	8	6	02/29/2020 07:21:08	02/29/2020 07:22:06	58	Recommendation List
13	1	1	03/29/2020 22:48:47	03/29/2020 22:52:04	163	Playlist
14	3	5	04/04/2020 11:00:05	04/04/2020 11:03:56	231	Search
15	3	9	04/20/2020 17:50:11	04/20/2020 17:54:13	242	Recommendation List
16	3	9	04/30/2020 00:00:21	04/30/2020 00:04:23	242	Playlist
17	8	9	05/02/2020 07:20:59	05/02/2020 07:22:01	62	Search

6. Where the User Accessed the Songs

```
SELECT q1.MUSIC_NUMBER,q1.MUSICNAME, LSOURCE
FROM MUSIC q1
INNER JOIN LISTEN q2 ON q1.MUSIC_NUMBER = q2.MUSIC_NUMBER
ORDER BY q1.MUSIC_NUMBER;
```

	MUSIC_NUMBER	MUSICNAME	LSOURCE
1	1	I Like Me Better	Search
2	1	I Like Me Better	Search
3	1	I Like Me Better	Recommendation List
4	1	I Like Me Better	Recommendation List
5	1	I Like Me Better	Recommendation List
6	1	I Like Me Better	Playlist
7	1	I Like Me Better	Recommendation List
8	1	I Like Me Better	Recommendation List
9	1	I Like Me Better	Recommendation List
10	1	I Like Me Better	Search
11	1	I Like Me Better	Recommendation List
12	2	Paris in the Rain	Recommendation List
13	2	Paris in the Rain	Search
14	2	Paris in the Rain	Other Recommendation
15	3	Head Above Water	Search
16	3	Head Above Water	Other Recommendation
17	3	Head Above Water	Search
18	3	Head Above Water	Search
19	4	I Fell In Love With The Devil	Other Recommendation
20	4	I Fell In Love With The Devil	Other Recommendation
21	4	I Fell In Love With The Devil	Search
22	4	I Fell In Love With The Devil	Recommendation List
23	4	I Fell In Love With The Devil	Recommendation List
24	4	I Fell In Love With The Devil	Recommendation List
25	5	Not a Lover	Playlist
26	5	Not a Lover	Playlist
27	5	Not a Lover	Playlist
28	5	Not a Lover	Playlist
29	5	Not a Lover	Other Recommendation
30	5	Not a Lover	Recommendation List

31	5	Not a Lover	Playlist
32	5	Not a Lover	Search
33	5	Not a Lover	Recommendation List
34	5	Not a Lover	Playlist
35	6	You Are	Search
36	6	You Are	Recommendation List
37	7	Okay	Other Recommendation
38	8	11887	Playlist
39	8	11887	Playlist
40	8	11887	Playlist
41	9	Not That Easy	Search
42	9	Not That Easy	Playlist
43	9	Not That Easy	Recommendation List
44	9	Not That Easy	Playlist
45	10	Second-Hand Soul	Playlist

7. The Performance of the Recommendation List

```
SELECT q4.PLAYLIST_NUMBER, q4.MUSIC_NUMBER, q4.FAVORITEDATE,
q4.ACCOUNTID, q3.LSOURCE
FROM LISTEN q3
INNER JOIN (SELECT q1.PLAYLIST_NUMBER, q1.MUSIC_NUMBER,
q1.FAVORITEDATE, q2.ACCOUNTID
FROM FAVORITE q1
INNER JOIN PLAYLIST q2 ON q1.PLAYLIST_NUMBER = q2.PLAYLIST_NUMBER) q4
ON q3.ACCOUNTID = q4.ACCOUNTID AND q3.MUSIC_NUMBER = q4.MUSIC_NUMBER
WHERE q3.LSOURCE = 'Recommendation List';
```

	PLAYLIST_NUMBER	MUSIC_NUMBER	FAVORITEDATE	ACCOUNTID	LSOURCE
1	1	1	10/04/2019	1	Recommendation List
2	1	1	10/04/2019	1	Recommendation List
3	2	5	12/21/2019	3	Recommendation List
4	10	9	04/30/2020	3	Recommendation List

8. Information of Premium Songs

```
SELECT *
```

FROM MUSIC

WHERE ISPREMIUM = 'Y';

	MUSIC_NUMBER	MUSICNAME	GENRE	ALBUM_NUMBER	SONGLENGTH	ISPREMIUM
1	3	Head Above Water	Pop Rock	3	220	Y
2	4	I Fell In Love With The Devil	Pop	3	255	Y
3	6	You Are	Pop	5	229	Y
4	7	Okay	Pop	5	205	Y
5	9	Not That Easy	Rock	7	242	Y

9. Songs' Genres Composition

SELECT GENRE, COUNT(GENRE) as GENRETOTAL

FROM MUSIC

GROUP BY GENRE;

	GENRE	GENRETOTAL
1	Pop	6
2	Rock	2
3	Pop Rock	1
4	R&B	1

10. Information of Our Songs Including Duration of the Copyright

SELECT q3.*, q2.albumname

FROM ALBUM q2

INNER JOIN (SELECT q1.MUSIC_NUMBER, MUSICNAME,ISSUEDATE,

ENDDATE,ENDDATE-ISSUEDATE as DurationOfCopyright, MUSIC.ALBUM_NUMBER

FROM Issuance_on_Platform q1

JOIN MUSIC ON q1.MUSIC_NUMBER = MUSIC.MUSIC_NUMBER

ORDER BY DurationOfCopyright DESC) q3 on q2.album_number = q3.ALBUM_NUMBER;

	MUSIC_NUMBER	MUSICNAME	ISSUEDATE	ENDDATE	DURATIONOFCOPYRIGHT	ALBUM_NUMBER	ALBUMNAME
1	5	Not a Lover	07/01/2019	07/01/2049	10958	4	Not a Lover
2	8	11887	06/12/2019	06/12/2049	10958	6	Dim Sum
3	7	Okay	06/06/2019	06/06/2049	10958	5	Silent Rebellion
4	6	You Are	06/06/2019	06/06/2049	10958	5	Silent Rebellion
5	3	Head Above Water	02/14/2019	02/14/2049	10958	3	Head Above Water
6	4	I Fell In Love With The Devil	02/14/2019	02/14/2049	10958	3	Head Above Water
7	9	Not That Easy	04/20/2020	04/20/2050	10957	7	2020
8	10	Second-Hand Soul	04/20/2020	04/20/2050	10957	7	2020
9	2	Paris in the Rain	11/16/2017	11/16/2047	10957	2	Paris in the Rain
10	1	I Like Me Better	05/18/2017	05/18/2047	10957	1	I Like Me Better

11. Members' Favorite Songs

```
SELECT q6.MUSIC_NUMBER, q5.MUSICNAME, q5.ISPREMIUM,
q6.PLAYLIST_NUMBER, q6.ACCOUNTID
FROM MUSIC q5
INNER JOIN (SELECT q3.MUSIC_NUMBER, q4.PLAYLIST_NUMBER, q4.ACCOUNTID
FROM FAVORITE q3
INNER JOIN (SELECT PLAYLIST_NUMBER, q2.ACCOUNTID
FROM PLAYLIST q1
INNER JOIN (SELECT ACCOUNTID
FROM User_Account
JOIN Normal_Account ON User_Account.ACCOUNTID = Normal_Account.NACCOUNTID
WHERE Normal_Account.ISMEMBER = 'Y') q2 on q1.ACCOUNTID = q2.ACCOUNTID) q4
ON q3.PLAYLIST_NUMBER = q4.PLAYLIST_NUMBER) q6
ON q5.MUSIC_NUMBER = q6.MUSIC_NUMBER;
```

	MUSIC_NUMBER	MUSICNAME	ISPREMIUM	PLAYLIST_NUMBER	ACCOUNTID
1	1	I Like Me Better	N	1	1
2	2	Paris in the Rain	N	3	2
3	4	I Fell In Love With The Devil	Y	2	3
4	5	Not a Lover	N	8	8
5	5	Not a Lover	N	2	3
6	9	Not That Easy	Y	10	3

12. Listening Duration

```
SELECT ACCOUNTID, q1.MUSIC_NUMBER, q2.GENRE,
    SUBSTR((StopTime-StartTime), INSTR((StopTime-StartTime),' ')+7,2) "Second",
    SUBSTR((StopTime-StartTime), INSTR((StopTime-StartTime),' ')+4,2) "Minute",
    SUBSTR((StopTime-StartTime), INSTR((StopTime-StartTime),' ')+1,2) "Hour"
```

FROM Listen q1

INNER JOIN (SELECT GENRE, MUSIC_NUMBER FROM MUSIC) q2

on q1.MUSIC_NUMBER = q2.MUSIC_NUMBER;

	ACCOUNTID	MUSIC_NUMBER	GENRE	Second	Minute	Hour
1	1	2	Pop	00	01	00
2	1	1	Pop	17	03	00
3	1	1	Pop	43	02	00
4	3	4	Pop	20	03	00
5	3	1	Pop	17	03	00
6	3	1	Pop	43	01	00
7	3	5	R&B	51	03	00
8	2	3	Pop Rock	40	03	00
9	2	2	Pop	15	04	00
10	6	3	Pop Rock	40	03	00
11	6	3	Pop Rock	50	02	00
12	7	8	Pop	32	04	00
13	7	8	Pop	32	04	00
14	7	8	Pop	32	04	00
15	8	4	Pop	14	02	00
16	9	5	R&B	25	00	00
17	7	1	Pop	17	03	00
18	7	1	Pop	17	03	00
19	7	1	Pop	17	03	00
20	7	1	Pop	17	03	00
21	7	1	Pop	17	02	00
22	12	3	Pop Rock	41	00	00
23	10	5	R&B	51	03	00
24	10	5	R&B	51	03	00
25	10	5	R&B	51	03	00
26	10	5	R&B	51	03	00
27	10	5	R&B	51	03	00
28	11	4	Pop	15	04	00
29	11	4	Pop	15	04	00
30	11	4	Pop	15	04	00

31	12	7	Pop	25	03	00
32	8	5	R&B	51	03	00
33	8	6	Pop	58	00	00
34	9	2	Pop	11	02	00
35	7	5	R&B	51	03	00
36	1	1	Pop	17	03	00
37	3	5	R&B	51	03	00
38	7	1	Pop	17	03	00
39	5	6	Pop	49	03	00
40	11	4	Pop	15	04	00
41	7	10	Rock	59	03	00
42	3	9	Rock	02	04	00
43	11	9	Rock	02	04	00
44	3	9	Rock	02	04	00
45	8	9	Rock	02	01	00

13. Count the Number of Times A Single Song is Liked by Users

```
SELECT q3.MUSIC_NUMBER, q4.MUSICNAME, q3.NUMBEROFLIKE
FROM MUSIC q4
INNER JOIN
(SELECT MUSIC_NUMBER, COUNT(ATTITUDE)as NumberOfLike
FROM (SELECT q1.MUSIC_NUMBER, q2.ATTITUDE
FROM MUSIC q1
INNER JOIN (SELECT MUSIC_NUMBER, ATTITUDE FROM Reflection) q2
ON q1.MUSIC_NUMBER = q2.MUSIC_NUMBER
WHERE ATTITUDE = 'Like')
GROUP BY MUSIC_NUMBER) q3 ON q4.MUSIC_NUMBER = q3.MUSIC_NUMBER;
```

	MUSIC_NUMBER	MUSICNAME	NUMBEROFLIKE
1	1	I Like Me Better	1
2	2	Paris in the Rain	1
3	5	Not a Lover	3
4	4	I Fell In Love With The Devil	2
5	8	11887	1
6	3	Head Above Water	1
7	9	Not That Easy	1

14. Sum Listening Percentage of Each Song

```
SELECT q5.MUSIC_NUMBER, q5.MUSICNAME, q5.GENRE, q5.TOTALPERCENTAGE,
q6.TOTALTIMES
FROM (SELECT q3.MUSIC_NUMBER, q4.MUSICNAME, q4.GENRE,
ROUND(q3.TOTALSTOPPOINT/TOTALSONGLENGTH,3) as TOTALPERCENTAGE
FROM (SELECT q1.MUSIC_NUMBER, SUM(q1.SONGLENGTH) as
TOTALSONGLENGTH, SUM(q2.STOPPOINT) as TOTALSTOPPOINT
FROM MUSIC q1
INNER JOIN LISTEN q2 ON q1.MUSIC_NUMBER = q2.MUSIC_NUMBER
GROUP BY q1.MUSIC_NUMBER) q3
INNER JOIN MUSIC q4 ON q3.MUSIC_NUMBER = q4.MUSIC_NUMBER) q5
INNER JOIN (SELECT q1.MUSIC_NUMBER, COUNT(q2.MUSIC_NUMBER) as
TOTALTIMES
FROM MUSIC q1
INNER JOIN LISTEN q2 ON q1.MUSIC_NUMBER = q2.MUSIC_NUMBER
GROUP BY q1.MUSIC_NUMBER) q6 on q5.MUSIC_NUMBER = q6.MUSIC_NUMBER
ORDER BY TOTALPERCENTAGE DESC, TOTALTIMES DESC;
```

	MUSIC_NUMBER	MUSICNAME	GENRE	TOTALPERCENTAGE	TOTALTIMES
1	8	11887	Pop	1	3
2	10	Second-Hand Soul	Rock	1	1
3	7	Okay	Pop	1	1
4	4	I Fell In Love With The Devil	Pop	0.979	6
5	5	Not a Lover	R&B	0.9	10
6	9	Not That Easy	Rock	0.814	4
7	1	I Like Me Better	Pop	0.803	11
8	2	Paris in the Rain	Pop	0.788	3
9	3	Head Above Water	Pop Rock	0.74	4
10	6	You Are	Pop	0.627	2

15. Monthly Royalties Paid to Artists

```
SELECT q2.*, ROYALTY, TOTALPLAYED*q3.ROYALTY as Rewards
FROM (SELECT q1.* , COUNT(MUSIC_NUMBER) as TotalPlayed
FROM (SELECT MUSIC_NUMBER
FROM LISTEN
```

```
WHERE STARTTIME >= '01/01/2019' AND STARTTIME <= '12/31/2019' AND
STOPPOINT >= 30) q1
GROUP BY q1.MUSIC_NUMBER) q2
INNER JOIN ISSUANCE_ON_PLATFORM q3 ON q2.MUSIC_NUMBER =
q3.MUSIC_NUMBER;
```

	MUSIC_NUMBER	TOTALPLAYED	ROYALTY	REWARDS
1	1	7	0.003	0.021
2	2	1	0.003	0.003
3	4	4	0.006	0.024
4	5	5	0.001	0.005
5	8	3	0.001	0.003
6	3	1	0.006	0.006

PL/SQL and Results

1. **Use Percentage to See if the User Likes the Song**

```
CREATE OR REPLACE PROCEDURE USERPREFQUANT(StopPoint IN
NUMBER,SongLength IN NUMBER)
IS
BEGIN
 IF  ROUND(StopPoint/SongLength, 3) >= 0.50 THEN
    dbms_output.put_line( 'The user likes the song.');
 ELSE
    dbms_output.put_line( 'The user does not like the song.');
 END IF;
END;


--The result will be 'The user does not like the song'
EXEC USERPREFQUANT(10,30);
--The result will be 'The user likes the song'
EXEC USERPREFQUANT(200,320);
```

```
The user does not like the song.


PL/SQL procedure successfully completed.

The user likes the song.


PL/SQL procedure successfully completed.
```

2. Judge Membership Type using the MEMBERSHIP Function

```
CREATE OR REPLACE FUNCTION MEMBERSHIP(Order_Membership_Duration IN
NUMBER)
RETURN VARCHAR
IS
BEGIN
   CASE Order_Membership_Duration
   WHEN 1 THEN
   RETURN('Monthly');
   WHEN 6 THEN
   RETURN('Half-Year');
   WHEN 12 THEN
   RETURN('Whole-Year');
   END CASE;
END;

--Monthly
DECLARE MEMBERSHIPRESULT VARCHAR(250);
BEGIN
   MEMBERSHIPRESULT := MEMBERSHIP(1);
   dbms_output.put_line( MEMBERSHIPRESULT);
END;

--Whole-Year
```

```
DECLARE MEMBERSHIPRESULT VARCHAR(250);
BEGIN
   MEMBERSHIPRESULT := MEMBERSHIP(12);
   dbms_output.put_line( MEMBERSHIPRESULT);
END;

--Half-Year
DECLARE MEMBERSHIPRESULT VARCHAR(250);
BEGIN
   MEMBERSHIPRESULT := MEMBERSHIP(6);
   dbms_output.put_line( MEMBERSHIPRESULT);
        END;
```

```
Monthly

PL/SQL procedure successfully completed.

Whole-Year

PL/SQL procedure successfully completed.

Half-Year

PL/SQL procedure successfully completed.
```

3. If A User Listens to A Song Because of Our Recommendation List

```
CREATE OR REPLACE FUNCTION RECO(reco IN VARCHAR)
RETURN VARCHAR
IS
BEGIN
   IF reco = 'Recommendation List' THEN RETURN('The user listend to this song because of recommendation list.');
   ELSE RETURN('The user listened to this song through other sources.');
   END IF;
END;
```

```
--The result will be 'The user listend to this song because of recommendation list'
DECLARE
   ANSWER VARCHAR(250);
BEGIN
   ANSWER := RECO('Recommendation List');
   dbms_output.put_line(ANSWER);
END;
--The result will be 'The user listened to this song through other sources'
DECLARE
   ANSWER VARCHAR(250);
BEGIN
   ANSWER := RECO('Search');
   dbms_output.put_line(ANSWER);
END;
```

```
The user listend to this song because of recommendation list.


PL/SQL procedure successfully completed.

The user listened to this song through other sources.


PL/SQL procedure successfully completed.
```

4. Our Users' Genders and Ages

```
CREATE OR REPLACE FUNCTION GENDERAGE(Gender IN CHAR, Age IN NUMBER)
RETURN VARCHAR
IS
BEGIN
   IF (Gender = 'M') AND (Age < 18) THEN RETURN('The user is a boy.');
   ELSIF (Gender = 'F') AND (Age < 18) THEN RETURN('The user is a girl.');
   ELSIF (Gender = 'M') AND (Age >= 18) THEN RETURN('The user is a gentleman.');
   ELSE RETURN('The user is a lady.');
   END IF;
```

```
END;

--The result will be 'The user is a boy'
DECLARE
   ANSWER VARCHAR(250);
BEGIN
   ANSWER := GENDERAGE('M',16);
   dbms_output.put_line(ANSWER);
END;

--The result will be 'The user is a lady'
DECLARE
   ANSWER VARCHAR(250);
BEGIN
   ANSWER := GENDERAGE('F',40);
   dbms_output.put_line(ANSWER);
END;

--The result will be 'The user is a girl'
DECLARE
   ANSWER VARCHAR(250);
BEGIN
   ANSWER := GENDERAGE('F',6);
   dbms_output.put_line(ANSWER);
END;
```

```
The user is a boy.

PL/SQL procedure successfully completed.

The user is a lady.

PL/SQL procedure successfully completed.

The user is a girl.

PL/SQL procedure successfully completed.

The user is a gentleman.

PL/SQL procedure successfully completed.
```

5. Users' Attitudes to the Songs

```
CREATE OR REPLACE PROCEDURE ATTITUDE(Attitude IN VARCHAR)
IS
BEGIN
   If Attitude = 'Like' THEN dbms_output.put_line( 'The user likes the song.');
   ELSE dbms_output.put_line( 'The user does not like the song.');
   END IF;
END;

--The result should be 'The user likes the song'
EXEC ATTITUDE('Like');
-- The result should be 'The user does not like the song'
EXEC ATTITUDE('Dislike');
```

```
The user likes the song.

PL/SQL procedure successfully completed.

The user does not like the song.

PL/SQL procedure successfully completed.
```

Additional Features Implemented

1. Print if the User Listened to A Song Because of the Recommendation List

Combine our RECO function and selection from table to see if the user listened to the song because of our recommendation list or not.

```
BEGIN
  FOR c IN
  --Select our table
  (SELECT q3.USERNAME, q3.LSOURCE, q4.MUSICNAME FROM (SELECT
q1.USERNAME, q2.LSOURCE, q2.MUSIC_NUMBER FROM User_Account q1
  INNER JOIN (SELECT ACCOUNTID, LSOURCE, MUSIC_NUMBER FROM LISTEN) q2
ON q1.ACCOUNTID = q2.ACCOUNTID) q3
  INNER JOIN MUSIC q4 ON q3.MUSIC_NUMBER = q4.MUSIC_NUMBER)
  --For loop
  LOOP
      --Using if statement to generate different answers for using recommendation list or not
      IF c.LSOURCE = 'Recommendation List' THEN dbms_output.put_line(c.USERNAME || '
listened to <' || c.MUSICNAME || '> and '||LOWER(RECO(c.LSOURCE)));
      ELSE dbms_output.put_line(c.USERNAME || ' listened to <' || c.MUSICNAME || '> but
'||LOWER(RECO(c.LSOURCE)));
      END IF;
  END LOOP;
END;
```

```
Adam listened to <I Like Me Better> but the user listened to this song through other sources.
Adam listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Adam listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Adam listened to <Paris in the Rain> and the user listend to this song because of recommendation list.
Ray listened to <Paris in the Rain> but the user listened to this song through other sources.
Ray listened to <Head Above Water> but the user listened to this song through other sources.
Cassie listened to <Not That Easy> but the user listened to this song through other sources.
Cassie listened to <Not That Easy> and the user listend to this song because of recommendation list.
Cassie listened to <Not a Lover> but the user listened to this song through other sources.
Cassie listened to <Not a Lover> and the user listend to this song because of recommendation list.
Cassie listened to <I Like Me Better> but the user listened to this song through other sources.
Cassie listened to <I Like Me Better> but the user listened to this song through other sources.
Cassie listened to <I Fell In Love With The Devil> but the user listened to this song through other sources.
Bu Cai listened to <You Are> but the user listened to this song through other sources.
Lauv listened to <Head Above Water> but the user listened to this song through other sources.
Lauv listened to <Head Above Water> but the user listened to this song through other sources.
Xuesi listened to <Second-Hand Soul> but the user listened to this song through other sources.
Xuesi listened to <I Like Me Better> but the user listened to this song through other sources.
Xuesi listened to <Not a Lover> and the user listend to this song because of recommendation list.
Xuesi listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Xuesi listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Xuesi listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Xuesi listened to <I Like Me Better> and the user listend to this song because of recommendation list.
Xuesi listened to <I Like Me Better> and the user listend to this song because of recommendation list.
```

```
Xuesi listened to <11887> but the user listened to this song through other sources.
Xuesi listened to <11887> but the user listened to this song through other sources.
Xuesi listened to <11887> but the user listened to this song through other sources.
Simon listened to <Not That Easy> but the user listened to this song through other sources.
Simon listened to <You Are> and the user listend to this song because of recommendation list.
Simon listened to <Not a Lover> but the user listened to this song through other sources.
Simon listened to <I Fell In Love With The Devil> but the user listened to this song through other sources.
Rosalie listened to <Paris in the Rain> but the user listened to this song through other sources.
Rosalie listened to <Not a Lover> but the user listened to this song through other sources.
Hui Tian listened to <Not a Lover> but the user listened to this song through other sources.
Hui Tian listened to <Not a Lover> but the user listened to this song through other sources.
Hui Tian listened to <Not a Lover> but the user listened to this song through other sources.
Hui Tian listened to <Not a Lover> but the user listened to this song through other sources.
Hui Tian listened to <Not a Lover> but the user listened to this song through other sources.
Chunhua Xie listened to <Not That Easy> but the user listened to this song through other sources.
Chunhua Xie listened to <I Fell In Love With The Devil> but the user listened to this song through other sources.
Chunhua Xie listened to <I Fell In Love With The Devil> and the user listend to this song because of recommendation list.
Chunhua Xie listened to <I Fell In Love With The Devil> and the user listend to this song because of recommendation list.
Chunhua Xie listened to <I Fell In Love With The Devil> and the user listend to this song because of recommendation list.
Feng Wang listened to <Okay> but the user listened to this song through other sources.
Feng Wang listened to <Head Above Water> but the user listened to this song through other sources.


PL/SQL procedure successfully completed.
```

2. Print Information of the Membership Payment

Combine our MEMBERSHIP function and selection from table to generate a running account about membership purchases.

```
BEGIN
  FOR c IN
  --Select our table
```

```
   (SELECT q2.USERNAME, q1.MEMBERSHIP_DURATION, q1.PAYMENTDATE FROM
(SELECT NACCOUNTID, MEMBERSHIP_DURATION, PAYMENTDATE FROM
MEMBERSHIP_PAYMENT) q1
   INNER JOIN User_Account q2 ON q1.NACCOUNTID = q2.ACCOUNTID ORDER BY
q1.PAYMENTDATE DESC)
   --For loop
   LOOP
      --Generate the running account about membership purchases
      dbms_output.put_line(c.USERNAME||' bought a
'||MEMBERSHIP(c.MEMBERSHIP_DURATION)||' membership on '||c.PAYMENTDATE||'.');
   END LOOP;
END;
```

```
Simon bought a Monthly membership on 05/08/2020.
Adam bought a Whole-Year membership on 05/02/2020.
Simon bought a Monthly membership on 04/07/2020.
Cassie bought a Whole-Year membership on 04/03/2020.
Ray bought a Half-Year membership on 12/07/2019.
Ray bought a Monthly membership on 11/06/2019.
Ray bought a Monthly membership on 10/05/2019.
Cassie bought a Half-Year membership on 10/02/2019.


PL/SQL procedure successfully completed.
```