Group: Apex DB

Team Members

Karuna Nadadur

Nerlande Lalanne

Qiuyin Yang

Siwen Yang

Project: ONLINE VIDEO STREAMING

Objectives

In a time where cable is almost obsolete, and most people subscribe to multiple streaming services, a lot of time is wasted browsing through ever-growing content across multiple subscriptions rather than consuming it given their busy schedules.

Our team, Apex DB, has a solution to this problem. We aim to make an application that consolidates movies and TV shows from different subscriptions and presents it to the user as a simple list that they would possibly want to watch. We plan to achieve this using information in the user profile on what kind of content they like to watch or have preferred previously. The application would also consider the relative ratings the movie or TV show has received so it can suggest a customized list of content that spans across all streaming services they currently subscribe to – allowing users to spend time getting entertained rather than searching for it.

User Requirements

(The descriptions of the various attributes are provided in table format later in the document.)

1. Every video stored in the APEX database is identified by a tile number, and further defined by a name, TV rating, IMDB rating, description, release date, run time and a thumbnail. There can be two types of titles.
    a. Movies – That includes but not limited to documentaries, trailers, movies, stand-up comedies.
    b. TV Shows – Videos that have multiple seasons/episodes. A TV Show is different from a movie in that it is defined by season number and episode number.
2. There can be several genres of titles and a genre is defined by a name.
3. A TV rating for every video is the TV parental guideline that is provided by Federal Communications Commission.

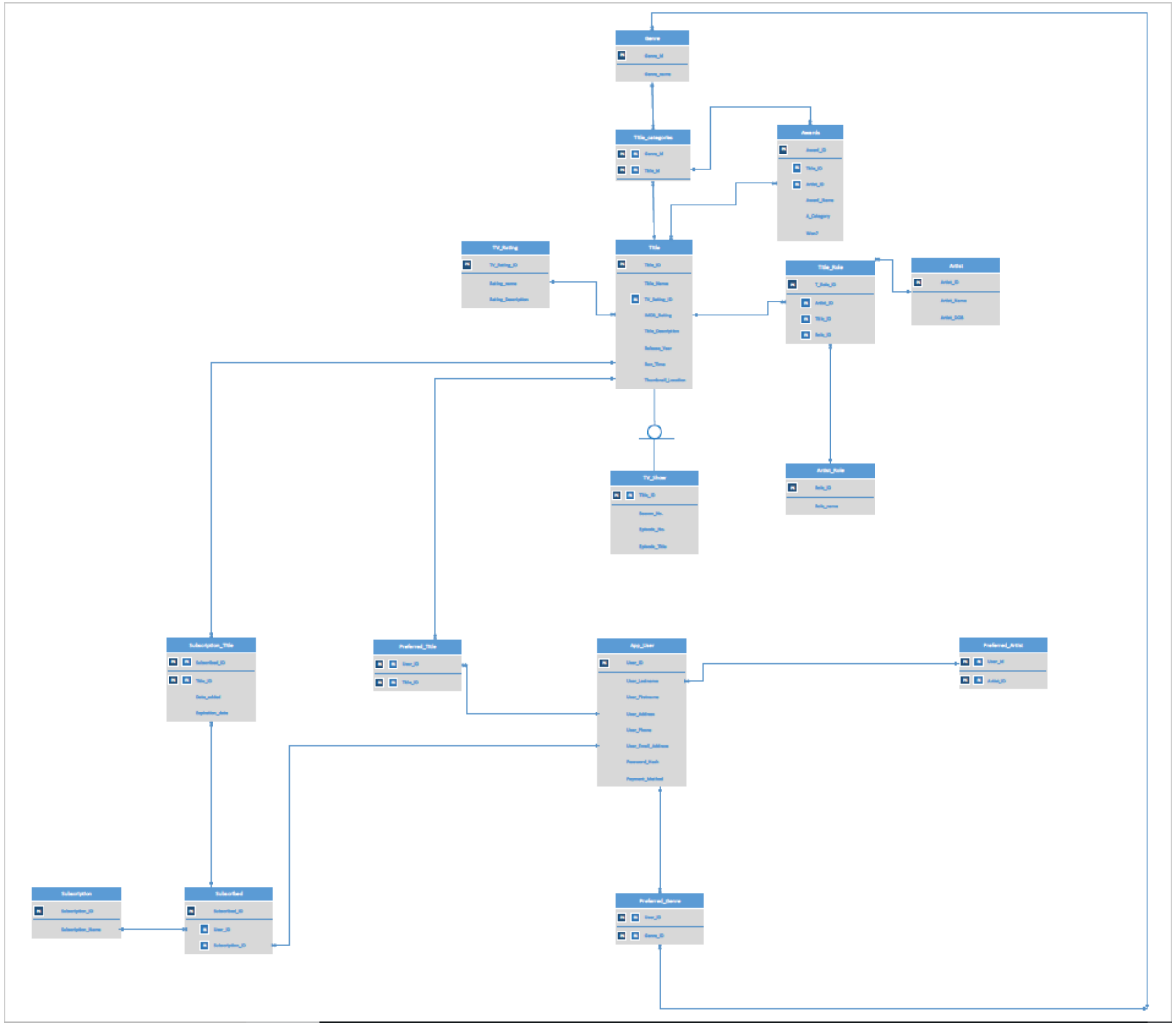
4. An artist can either be a cast member or a crew member. An artist is defined by a name and date of birth. An artist is also described by his/her screen name.

Business Rules

1. The database should provide provision for usage of a Thumbnail in the future.
2. A User can have multiple subscriptions.
3. The same video can be available in multiple subscriptions.
4. A subscription has an expiry date.
5. An award can either be won or just be nominated for.
6. IMDB ratings can be picked directly from the IMDB website.

Extended Entity Relationship Diagram

The EERD below reflects the above stated rule.



Relationship Definitions:

1. An artist can play one more roles for every title. And a title has many artists.
2. An artist may be nominated for many awards and multiple artists may receive the same award over a period.
3. A user is defined by a name and email ID. Apex stores the email ID and password hash for every user for login purposes.
4. A user may subscribe to multiple subscriptions. And a subscription may be subscribed by many users.
5. A user may have a preference of multiple genres. And one genre maybe preferred by multiple users.
6. A user may have a preference of multiple artists. And one artist maybe preferred by multiple users.
7. A user may have a preference of multiple titles. And one title maybe preferred by multiple users.

Tables

		Description
Table Name	**Title**	
Attributes	Title_ID (PK)	*Identifier*
	Title_Name	*Name of title*
	TV_Rating_ID	Rating provided by FCC
	IMDB_Rating	*IMDB rating issued*
	Title_Description	Short description of the video
	Release_Year	*Year of first release*
	Run_Time	*Length of video*
	Thumbnail_Location	*Picture*

		Description
Table Name	**TV_Show**	
Attributes	Title_ID(PK,FK)	*Identifier*
	Season_No.	*Season number*
	Episode_No.	*Episode number*
	Episode_Title	*Title of episode*

		Description
Table Name	**TV_Rating**	
Attributes	TV_Rating_ID(PK)	Rating provided by FCC
	Rating_Name	Ex: TV-14, TV-PG etc
	Rating_Description	*What the rating means*

		Description
Table Name	**Genre**	
Attributes	Genre_ID(PK)	Ex. Anime, Children, Horror etc.
	Genre_Name	*Name of genre*

		Description
Table Name	**Title_Categories**	
Attributes	Genre_ID(PK,FK)	*Genre Identifier*
	Title_ID(PK,FK)	*Title identifier*

		Description
Table Name	**Artist_Role**	
Attributes	Role_ID	*Identifier*
	Role_Name	Ex. Director, Actor, Producer etc.

		Description
Table Name	**Title_Role**	
Attributes	T_Role_ID(PK)	*Identifier*
	Artist_ID(FK)	*Artist identifier*
	Title_ID(FK)	*Title identifier*
	Role_ID(FK)	*Role Identifier*

		Description
Table Name	**Artist**	
Attributes	Artist_ID(PK)	*Identifier*
	Artist_Name	*Name of artist*
	Artist_DOB	*Date of birth of artist*

		Description
Table Name	**Awards**	
Attributes	Award_ID(PK)	*Identifier*
	Title_ID(FK)	*Title Identifier*
	Artist_ID(FK)	*Artist Identifier*
	Award_Name	*Name of award*

	A_Category	*Award category*
	Won?	*If the Award was won or just nominated*

		Description
Table Name	**App_User**	
Attributes	User_ID(PK)	*Identifier*
	User_name	*Name of user*
	Email	*Email ID*
	User_address	*Address of user*
	User_Phone	*Phone number*
	Password_Hash	One way Hash for login purposes.
	Payment_method	Method of payment

		Description
Table Name	**Subscribed**	
Attributes	Subscribed_ID(PK)	*Identifier*
	User_ID(FK)	*Identifier*
	Subscription_ID(FK)	*Identifier*

		Description
Table Name	**Subscription**	
Attributes	Subscription_ID(PK)	*Identifier*
	Subscription_Name	Ex. Netflix, Hulu etc.

		Description
Table Name	**Subsciription_Title**	
Attributes	Title_ID(PK,FK)	*Identifier*
	Subscribed_ID(PK,FK)	*Identifier*
	Date_Added	*Date video added to subscription*

	Expiration_Date	*Date video expires in subscription*

		Description
Table Name	**Preferred_Title**	
Attributes	User_ID(PK,FK)	*Identifier*
	Title_ID(PK,FK)	*Identifier*

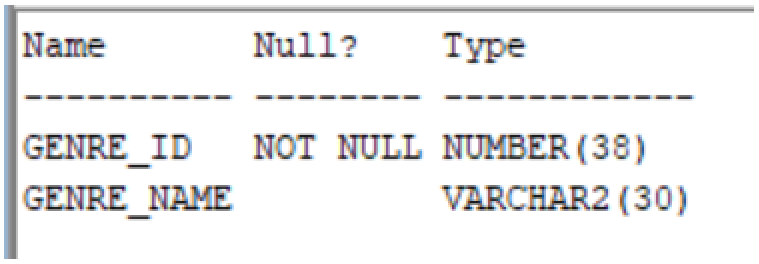
		Description
Table Name	**Preferred_Artist**	
Attributes	User_ID(PK,FK)	*Identifier*
	Artist_ID(PK,FK)	*Identifier*

		Description
Table Name	**Preferred_Genre**	
Attributes	User_ID(PK,FK)	*Identifier*
	Genre_ID(PK,FK)	*Identifier*

Code Screenshots

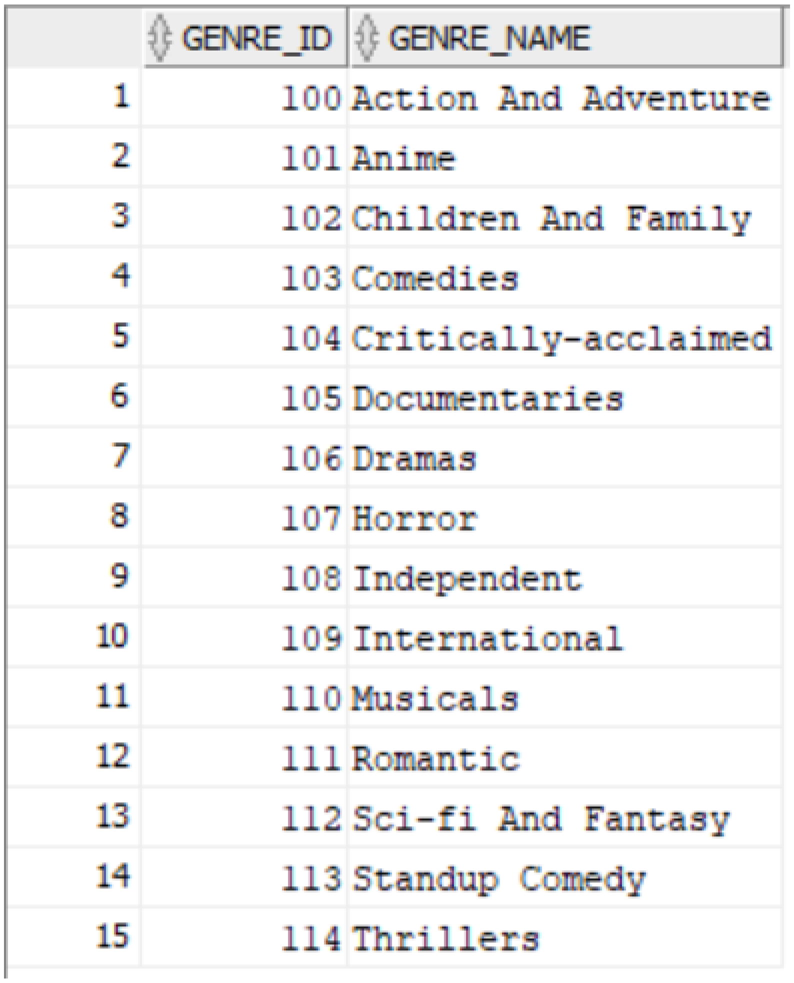
Table DESCRIBEs and SELECTs

DESCRIBE GENRE;



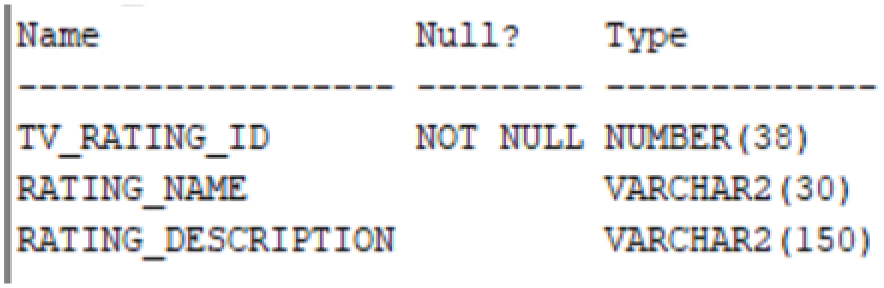
```
Name        Null?    Type
----------- -------- ------------
GENRE_ID    NOT NULL NUMBER(38)
GENRE_NAME           VARCHAR2(30)
```

SELECT * FROM GENRE;



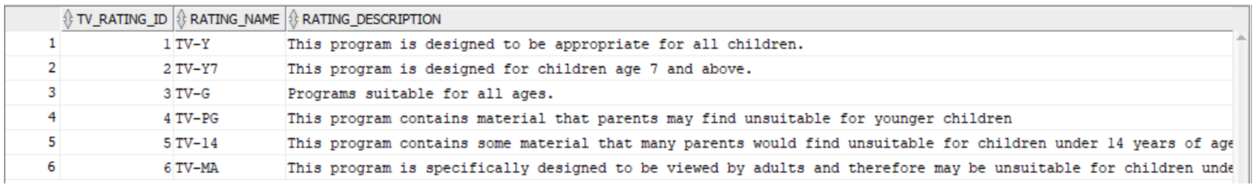
	GENRE_ID	GENRE_NAME
1	100	Action And Adventure
2	101	Anime
3	102	Children And Family
4	103	Comedies
5	104	Critically-acclaimed
6	105	Documentaries
7	106	Dramas
8	107	Horror
9	108	Independent
10	109	International
11	110	Musicals
12	111	Romantic
13	112	Sci-fi And Fantasy
14	113	Standup Comedy
15	114	Thrillers

DESCRIBE TV_RATING;



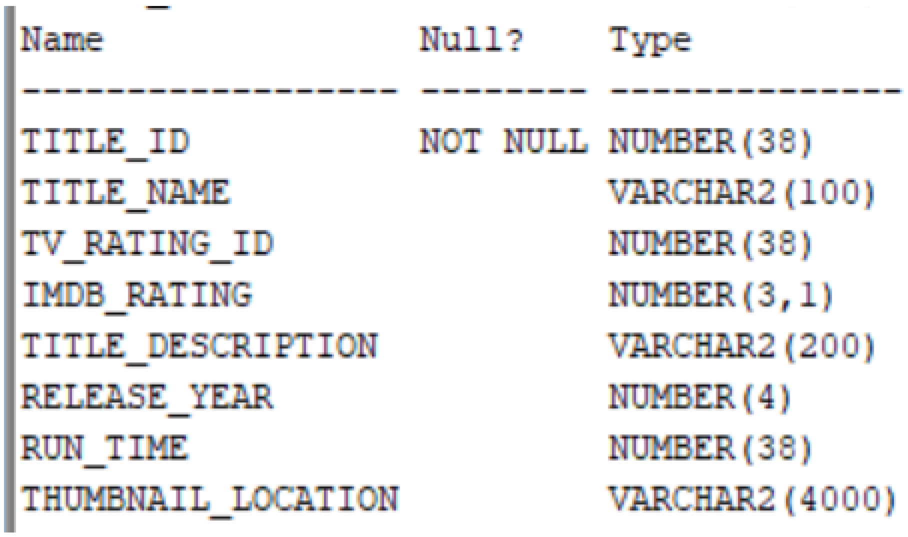
```
Name                  Null?    Type
------------------- -------- -------------
TV_RATING_ID         NOT NULL NUMBER(38)
RATING_NAME                   VARCHAR2(30)
RATING_DESCRIPTION            VARCHAR2(150)
```

SELECT * FROM TV_RATING;



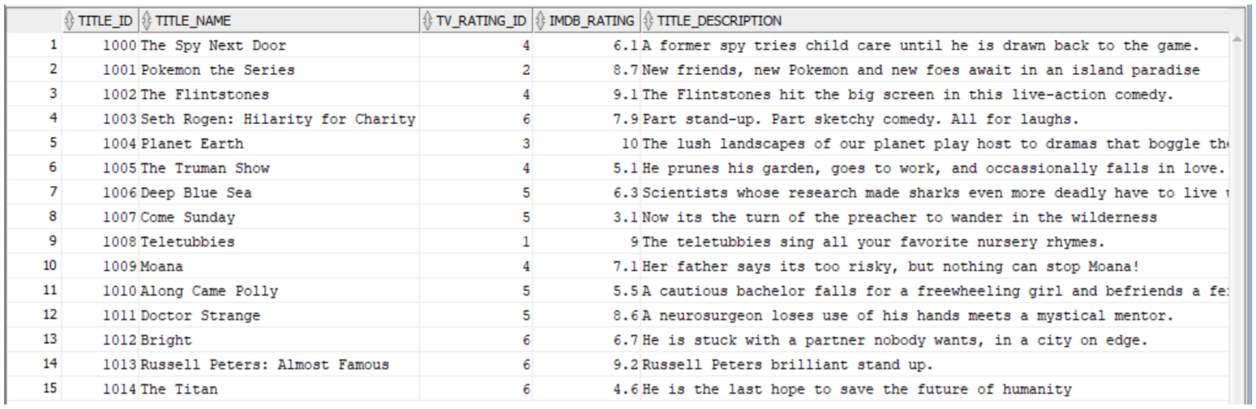
	TV_RATING_ID	RATING_NAME	RATING_DESCRIPTION
1	1	TV-Y	This program is designed to be appropriate for all children.
2	2	TV-Y7	This program is designed for children age 7 and above.
3	3	TV-G	Programs suitable for all ages.
4	4	TV-PG	This program contains material that parents may find unsuitable for younger children
5	5	TV-14	This program contains some material that many parents would find unsuitable for children under 14 years of age
6	6	TV-MA	This program is specifically designed to be viewed by adults and therefore may be unsuitable for children unde

DESCRIBE TITLE;



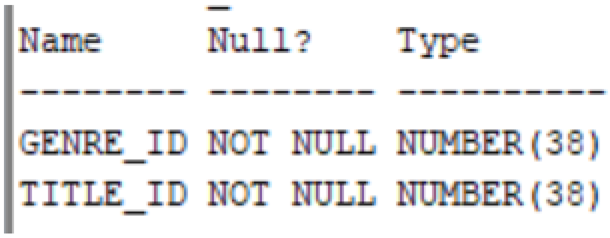
```
Name               Null?    Type
------------------ -------- -------------
TITLE_ID           NOT NULL NUMBER(38)
TITLE_NAME                  VARCHAR2(100)
TV_RATING_ID                NUMBER(38)
IMDB_RATING                 NUMBER(3,1)
TITLE_DESCRIPTION           VARCHAR2(200)
RELEASE_YEAR                NUMBER(4)
RUN_TIME                    NUMBER(38)
THUMBNAIL_LOCATION          VARCHAR2(4000)
```

SELECT * FROM TITLE;



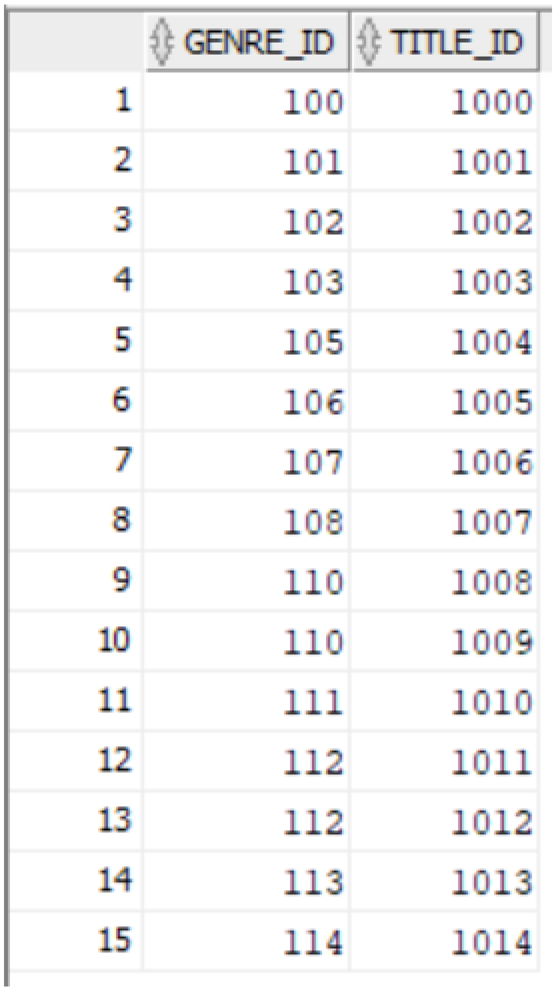
	TITLE_ID	TITLE_NAME	TV_RATING_ID	IMDB_RATING	TITLE_DESCRIPTION
1	1000	The Spy Next Door	4	6.1	A former spy tries child care until he is drawn back to the game.
2	1001	Pokemon the Series	2	8.7	New friends, new Pokemon and new foes await in an island paradise
3	1002	The Flintstones	4	9.1	The Flintstones hit the big screen in this live-action comedy.
4	1003	Seth Rogen: Hilarity for Charity	6	7.9	Part stand-up. Part sketchy comedy. All for laughs.
5	1004	Planet Earth	3	10	The lush landscapes of our planet play host to dramas that boggle th
6	1005	The Truman Show	4	5.1	He prunes his garden, goes to work, and occassionally falls in love.
7	1006	Deep Blue Sea	5	6.3	Scientists whose research made sharks even more deadly have to live
8	1007	Come Sunday	5	3.1	Now its the turn of the preacher to wander in the wilderness
9	1008	Teletubbies	1	9	The teletubbies sing all your favorite nursery rhymes.
10	1009	Moana	4	7.1	Her father says its too risky, but nothing can stop Moana!
11	1010	Along Came Polly	5	5.5	A cautious bachelor falls for a freewheeling girl and befriends a fe
12	1011	Doctor Strange	5	8.6	A neurosurgeon loses use of his hands meets a mystical mentor.
13	1012	Bright	6	6.7	He is stuck with a partner nobody wants, in a city on edge.
14	1013	Russell Peters: Almost Famous	6	9.2	Russell Peters brilliant stand up.
15	1014	The Titan	6	4.6	He is the last hope to save the future of humanity

DESCRIBE TITLE_CATEGORY;



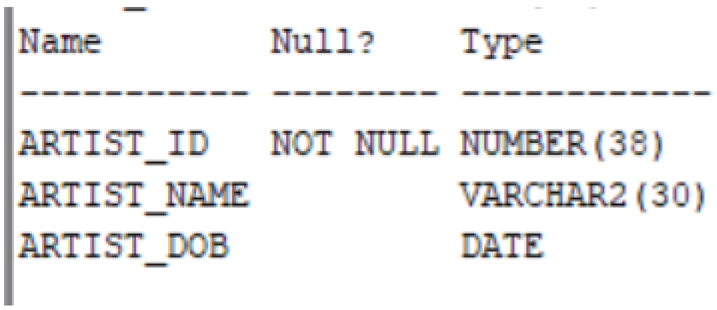
```
Name     Null?    Type
-------- -------- ----------
GENRE_ID NOT NULL NUMBER(38)
TITLE_ID NOT NULL NUMBER(38)
```

SELECT * FROM TITLE_CATEGORY;



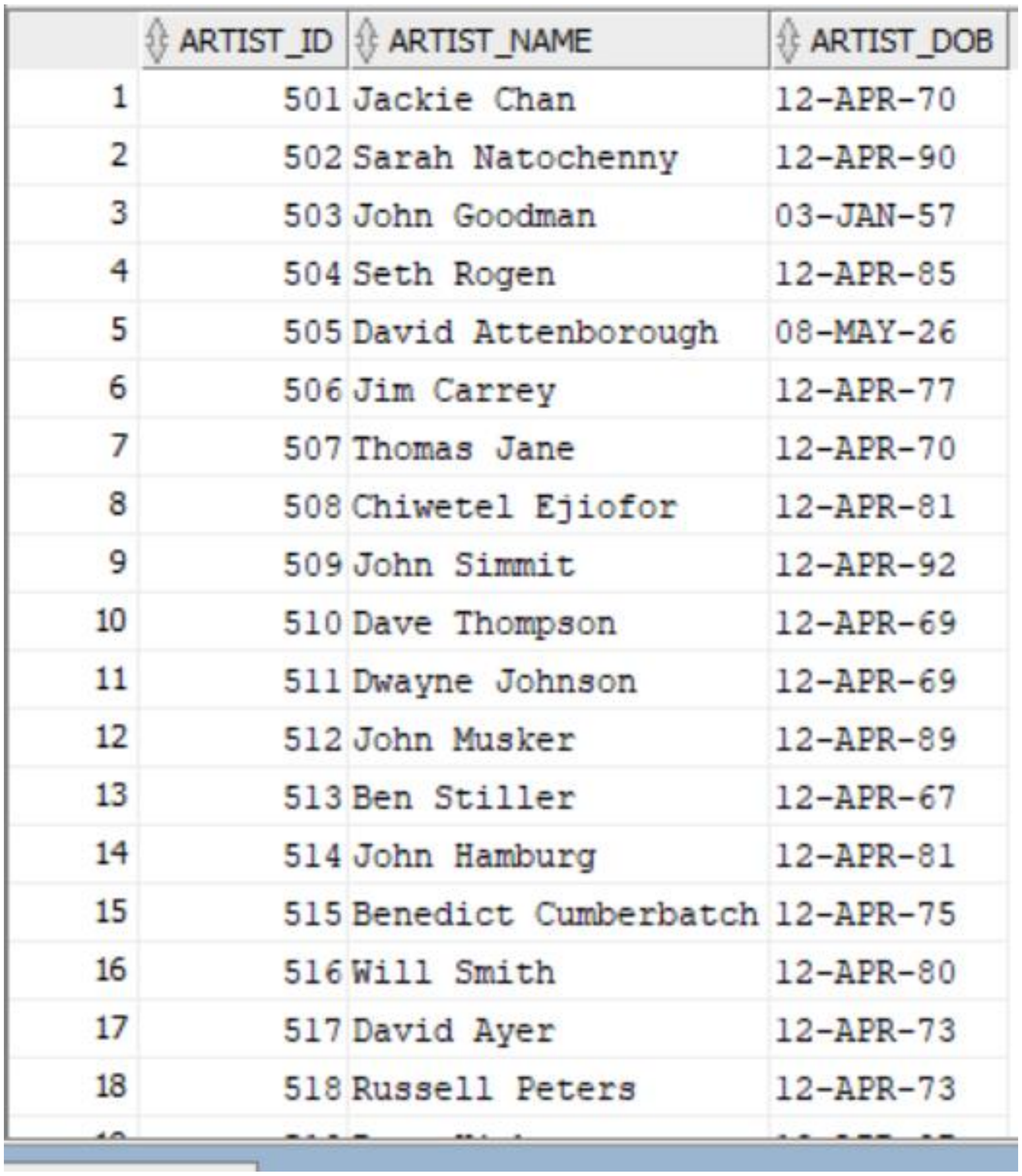
	GENRE_ID	TITLE_ID
1	100	1000
2	101	1001
3	102	1002
4	103	1003
5	105	1004
6	106	1005
7	107	1006
8	108	1007
9	110	1008
10	110	1009
11	111	1010
12	112	1011
13	112	1012
14	113	1013
15	114	1014

DESCRIBE ARTIST;



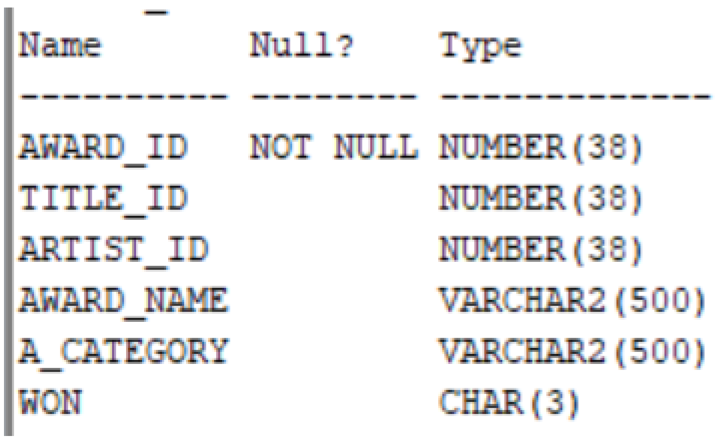
```
Name        Null?    Type
----------- -------- ------------
ARTIST_ID   NOT NULL NUMBER(38)
ARTIST_NAME          VARCHAR2(30)
ARTIST_DOB           DATE
```

SELECT * FROM ARTIST;



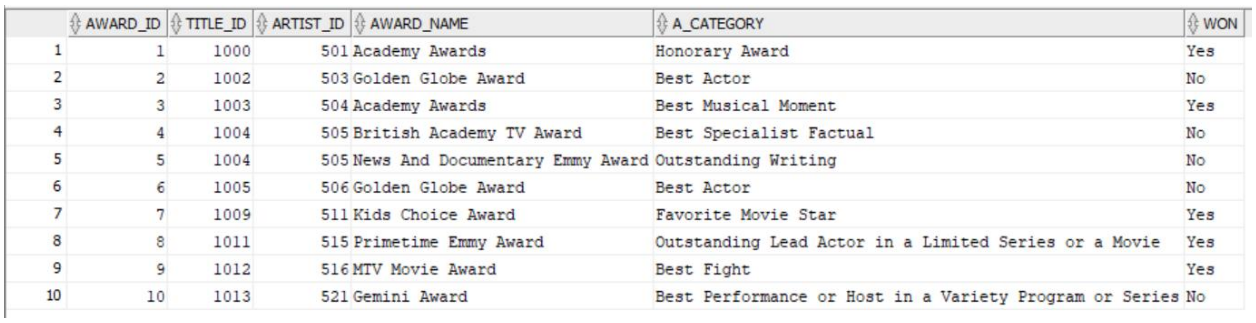
	ARTIST_ID	ARTIST_NAME	ARTIST_DOB
1	501	Jackie Chan	12-APR-70
2	502	Sarah Natochenny	12-APR-90
3	503	John Goodman	03-JAN-57
4	504	Seth Rogen	12-APR-85
5	505	David Attenborough	08-MAY-26
6	506	Jim Carrey	12-APR-77
7	507	Thomas Jane	12-APR-70
8	508	Chiwetel Ejiofor	12-APR-81
9	509	John Simmit	12-APR-92
10	510	Dave Thompson	12-APR-69
11	511	Dwayne Johnson	12-APR-69
12	512	John Musker	12-APR-89
13	513	Ben Stiller	12-APR-67
14	514	John Hamburg	12-APR-81
15	515	Benedict Cumberbatch	12-APR-75
16	516	Will Smith	12-APR-80
17	517	David Ayer	12-APR-73
18	518	Russell Peters	12-APR-73

DESCRIBE AWARD;



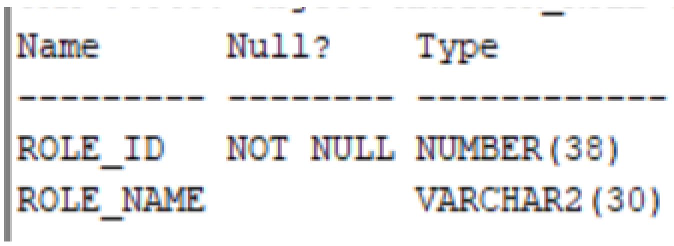
```
Name        Null?    Type
----------- -------- -------------
AWARD_ID    NOT NULL NUMBER(38)
TITLE_ID             NUMBER(38)
ARTIST_ID            NUMBER(38)
AWARD_NAME           VARCHAR2(500)
A_CATEGORY           VARCHAR2(500)
WON                  CHAR(3)
```

SELECT * FROM AWARD;



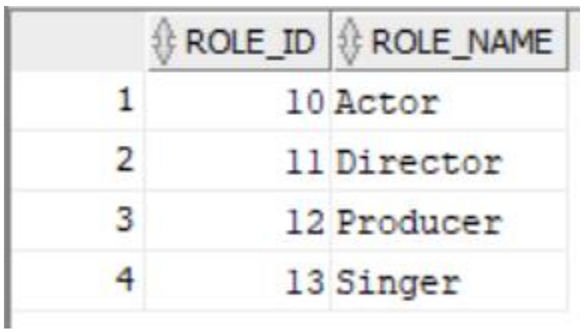
	AWARD_ID	TITLE_ID	ARTIST_ID	AWARD_NAME	A_CATEGORY	WON
1	1	1000	501	Academy Awards	Honorary Award	Yes
2	2	1002	503	Golden Globe Award	Best Actor	No
3	3	1003	504	Academy Awards	Best Musical Moment	Yes
4	4	1004	505	British Academy TV Award	Best Specialist Factual	No
5	5	1004	505	News And Documentary Emmy Award	Outstanding Writing	No
6	6	1005	506	Golden Globe Award	Best Actor	No
7	7	1009	511	Kids Choice Award	Favorite Movie Star	Yes
8	8	1011	515	Primetime Emmy Award	Outstanding Lead Actor in a Limited Series or a Movie	Yes
9	9	1012	516	MTV Movie Award	Best Fight	Yes
10	10	1013	521	Gemini Award	Best Performance or Host in a Variety Program or Series	No

DESCRIBE ARTISTR_ROLE;



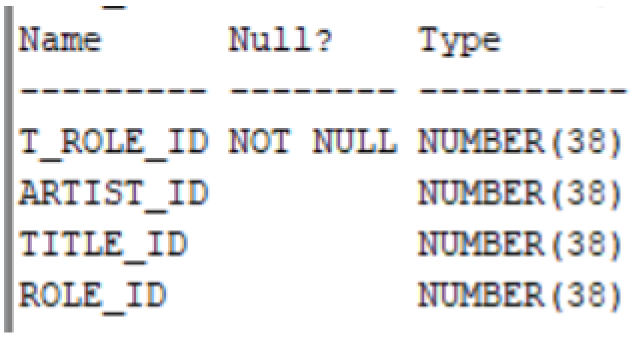
```
Name      Null?    Type
--------- -------- ------------
ROLE_ID   NOT NULL NUMBER(38)
ROLE_NAME          VARCHAR2(30)
```

SELECT * FROM ARTIST_ROLE;



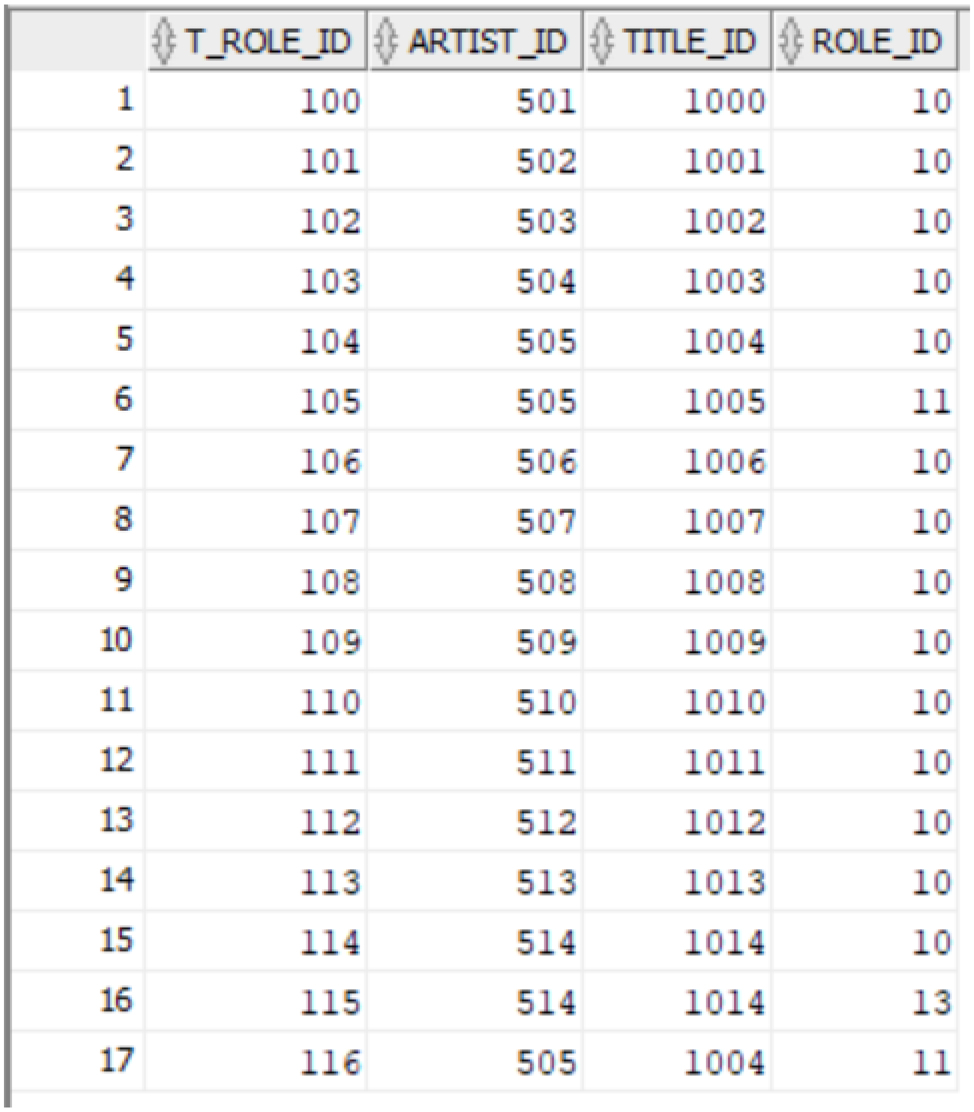
	ROLE_ID	ROLE_NAME
1	10	Actor
2	11	Director
3	12	Producer
4	13	Singer

DESCRIBE TITLE_ROLE;



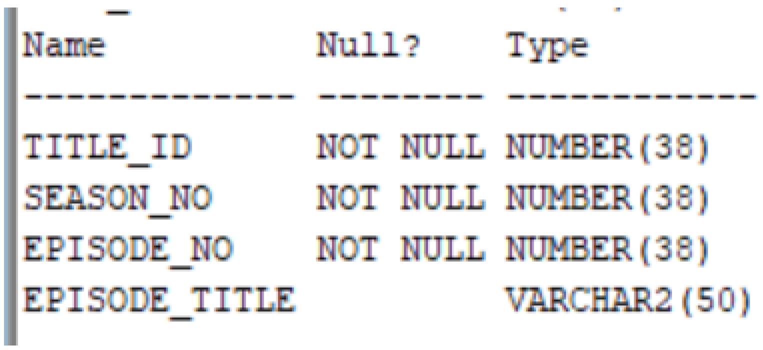
```
Name      Null?    Type
--------- -------- ----------
T_ROLE_ID NOT NULL NUMBER(38)
ARTIST_ID          NUMBER(38)
TITLE_ID           NUMBER(38)
ROLE_ID            NUMBER(38)
```

SELECT * FROM TITLE_ROLE;



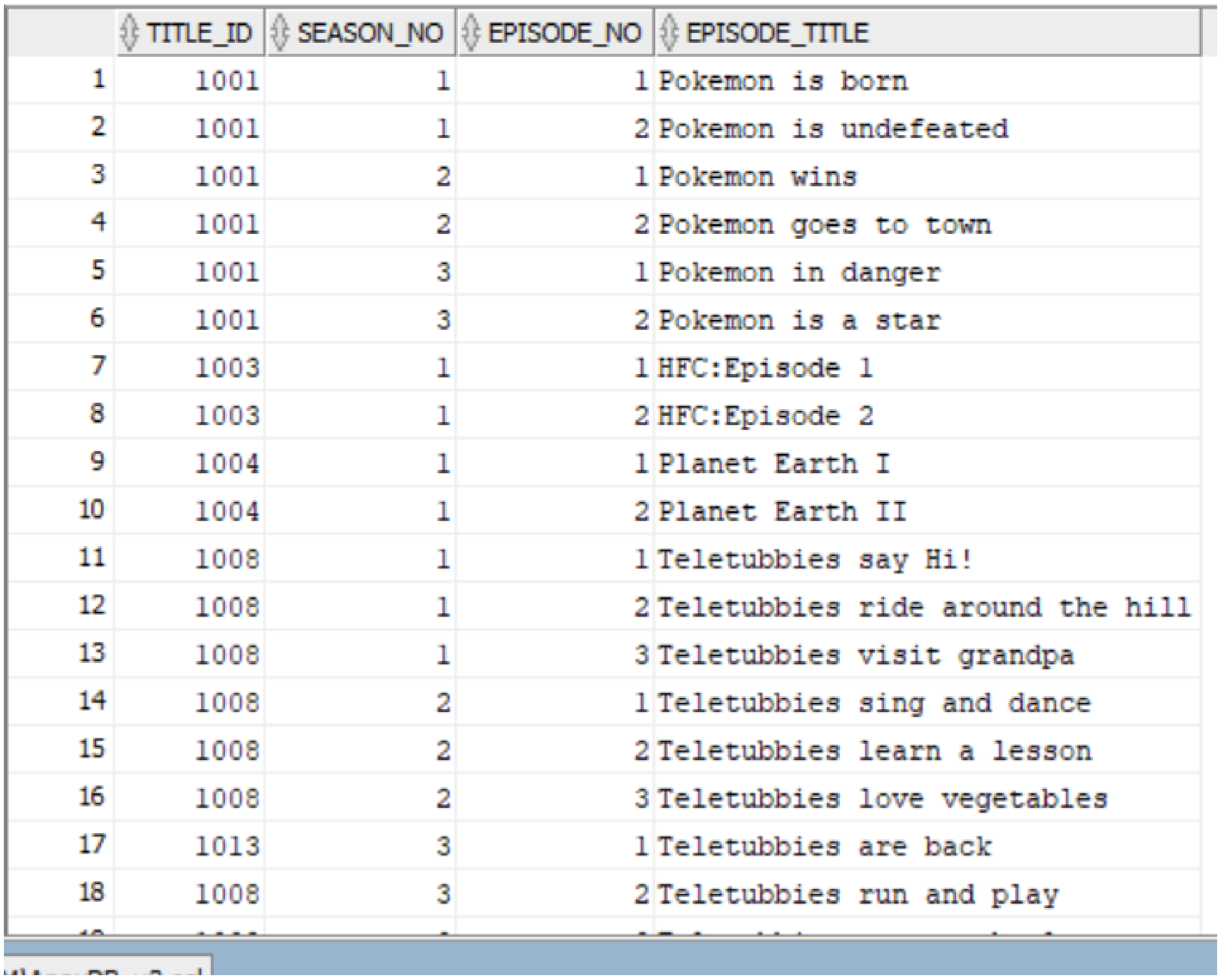
	T_ROLE_ID	ARTIST_ID	TITLE_ID	ROLE_ID
1	100	501	1000	10
2	101	502	1001	10
3	102	503	1002	10
4	103	504	1003	10
5	104	505	1004	10
6	105	505	1005	11
7	106	506	1006	10
8	107	507	1007	10
9	108	508	1008	10
10	109	509	1009	10
11	110	510	1010	10
12	111	511	1011	10
13	112	512	1012	10
14	113	513	1013	10
15	114	514	1014	10
16	115	514	1014	13
17	116	505	1004	11

DESCRIBE TV_SHOW;



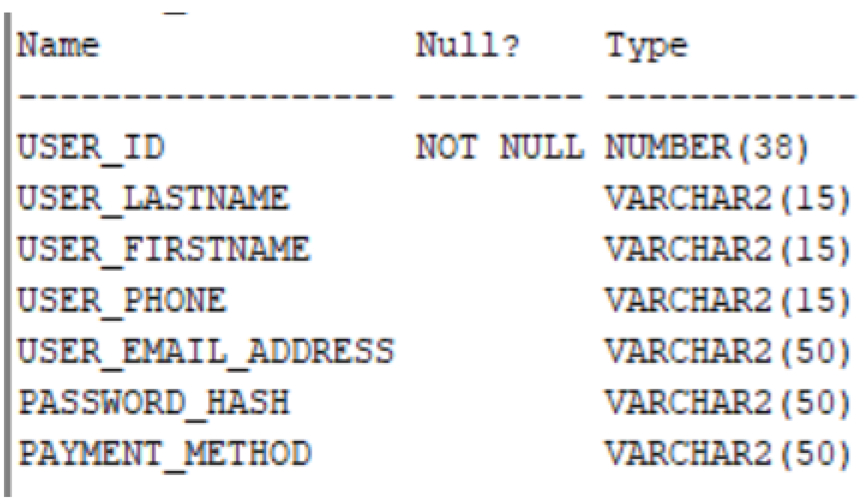
```
Name           Null?    Type
-------------- -------- ------------
TITLE_ID       NOT NULL NUMBER(38)
SEASON_NO      NOT NULL NUMBER(38)
EPISODE_NO     NOT NULL NUMBER(38)
EPISODE_TITLE           VARCHAR2(50)
```

SELECT * FROM TV_SHOW;



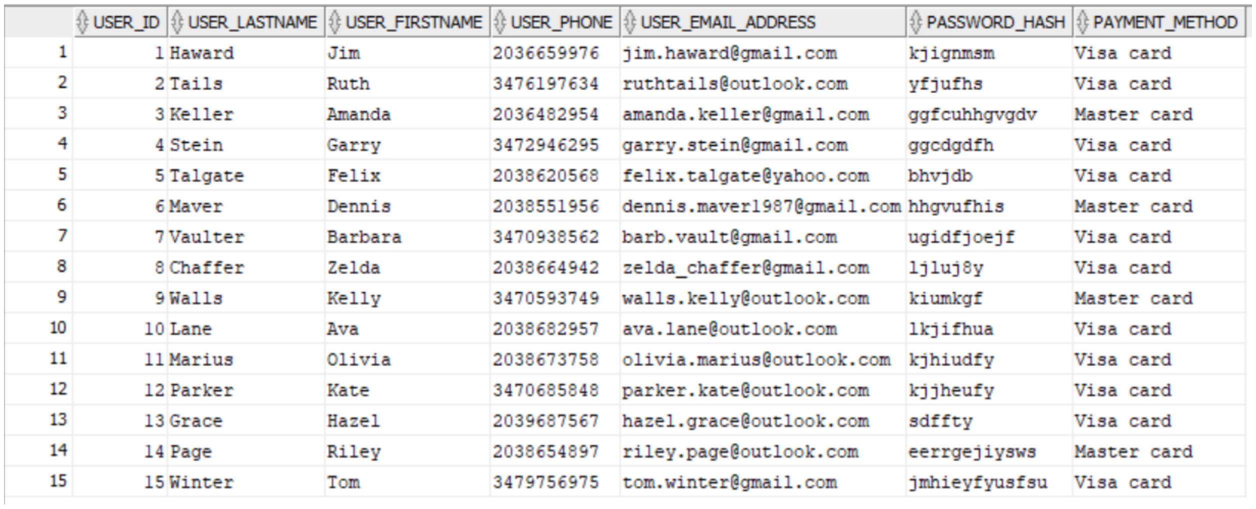
	TITLE_ID	SEASON_NO	EPISODE_NO	EPISODE_TITLE
1	1001	1	1	Pokemon is born
2	1001	1	2	Pokemon is undefeated
3	1001	2	1	Pokemon wins
4	1001	2	2	Pokemon goes to town
5	1001	3	1	Pokemon in danger
6	1001	3	2	Pokemon is a star
7	1003	1	1	HFC:Episode 1
8	1003	1	2	HFC:Episode 2
9	1004	1	1	Planet Earth I
10	1004	1	2	Planet Earth II
11	1008	1	1	Teletubbies say Hi!
12	1008	1	2	Teletubbies ride around the hill
13	1008	1	3	Teletubbies visit grandpa
14	1008	2	1	Teletubbies sing and dance
15	1008	2	2	Teletubbies learn a lesson
16	1008	2	3	Teletubbies love vegetables
17	1013	3	1	Teletubbies are back
18	1008	3	2	Teletubbies run and play

DESCRIBE APP_USER;



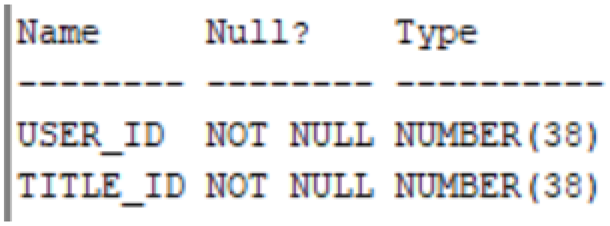
```
Name                Null?    Type
------------------- -------- ------------
USER_ID             NOT NULL NUMBER(38)
USER_LASTNAME                VARCHAR2(15)
USER_FIRSTNAME               VARCHAR2(15)
USER_PHONE                   VARCHAR2(15)
USER_EMAIL_ADDRESS           VARCHAR2(50)
PASSWORD_HASH                VARCHAR2(50)
PAYMENT_METHOD               VARCHAR2(50)
```

SELECT * FROM APP_USER;



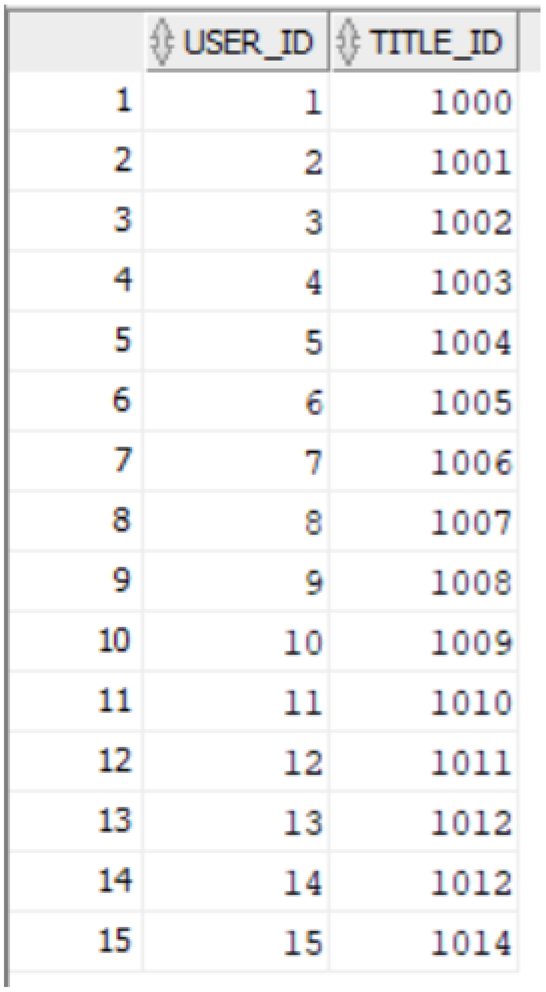
	USER_ID	USER_LASTNAME	USER_FIRSTNAME	USER_PHONE	USER_EMAIL_ADDRESS	PASSWORD_HASH	PAYMENT_METHOD
1	1	Haward	Jim	2036659976	jim.haward@gmail.com	kjignmsm	Visa card
2	2	Tails	Ruth	3476197634	ruthtails@outlook.com	yfjufhs	Visa card
3	3	Keller	Amanda	2036482954	amanda.keller@gmail.com	ggfcuhhgvgdv	Master card
4	4	Stein	Garry	3472946295	garry.stein@gmail.com	ggcdgdfh	Visa card
5	5	Talgate	Felix	2038620568	felix.talgate@yahoo.com	bhvjdb	Visa card
6	6	Maver	Dennis	2038551956	dennis.maver1987@gmail.com	hhgvufhis	Master card
7	7	Vaulter	Barbara	3470938562	barb.vault@gmail.com	ugidfjoejf	Visa card
8	8	Chaffer	Zelda	2038664942	zelda_chaffer@gmail.com	ljluj8y	Visa card
9	9	Walls	Kelly	3470593749	walls.kelly@outlook.com	kiumkgf	Master card
10	10	Lane	Ava	2038682957	ava.lane@outlook.com	lkjifhua	Visa card
11	11	Marius	Olivia	2038673758	olivia.marius@outlook.com	kjhiudfy	Visa card
12	12	Parker	Kate	3470685848	parker.kate@outlook.com	kjjheufy	Visa card
13	13	Grace	Hazel	2039687567	hazel.grace@outlook.com	sdffty	Visa card
14	14	Page	Riley	2038654897	riley.page@outlook.com	eerrgejiysws	Master card
15	15	Winter	Tom	3479756975	tom.winter@gmail.com	jmhieyfyusfsu	Visa card

DESCRIBE PREFERRED_TITLE;



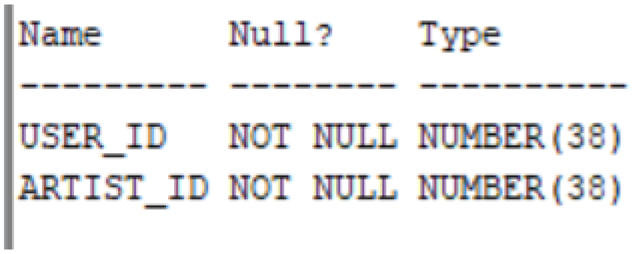
```
Name     Null?    Type
-------- -------- ----------
USER_ID  NOT NULL NUMBER(38)
TITLE_ID NOT NULL NUMBER(38)
```

SELECT * FROM PREFERRED_TITLE;



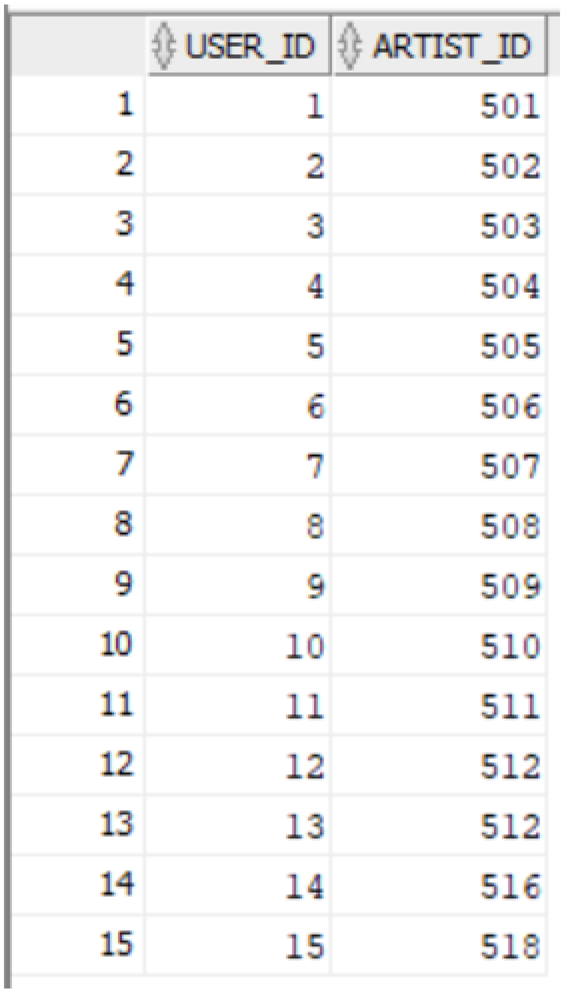
	USER_ID	TITLE_ID
1	1	1000
2	2	1001
3	3	1002
4	4	1003
5	5	1004
6	6	1005
7	7	1006
8	8	1007
9	9	1008
10	10	1009
11	11	1010
12	12	1011
13	13	1012
14	14	1012
15	15	1014

DESCRIBE PREFERRED_ARTIST;



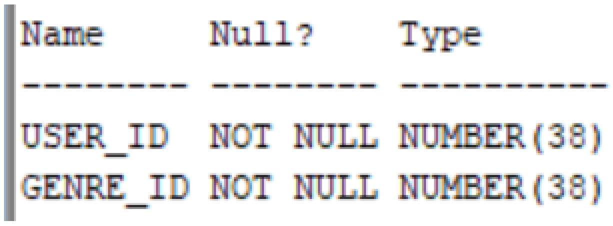
```
Name      Null?    Type
--------- -------- ----------
USER_ID   NOT NULL NUMBER(38)
ARTIST_ID NOT NULL NUMBER(38)
```

SELECT * FROM PREFERRED_ARTIST;



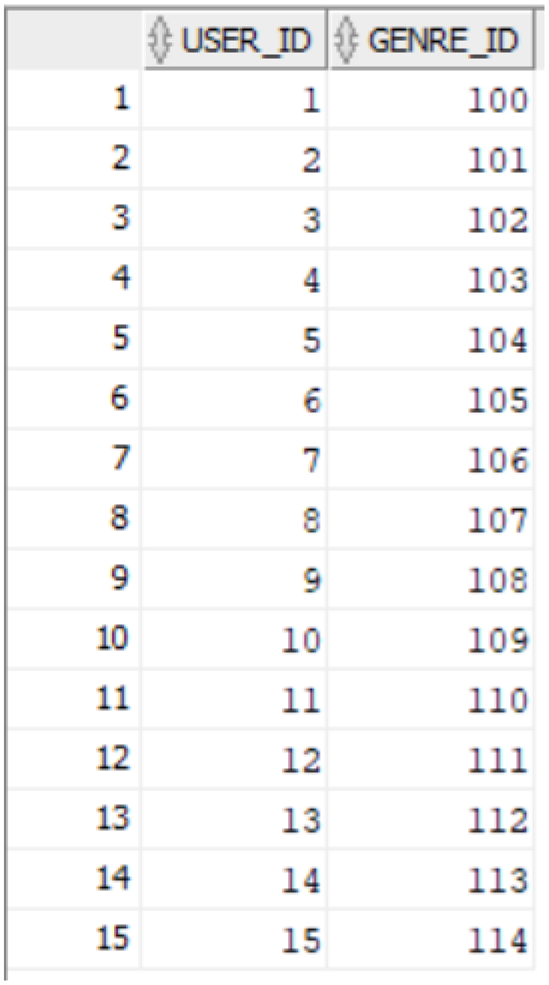
	USER_ID	ARTIST_ID
1	1	501
2	2	502
3	3	503
4	4	504
5	5	505
6	6	506
7	7	507
8	8	508
9	9	509
10	10	510
11	11	511
12	12	512
13	13	512
14	14	516
15	15	518

DESCRIBE PREFERRED_GENRE;



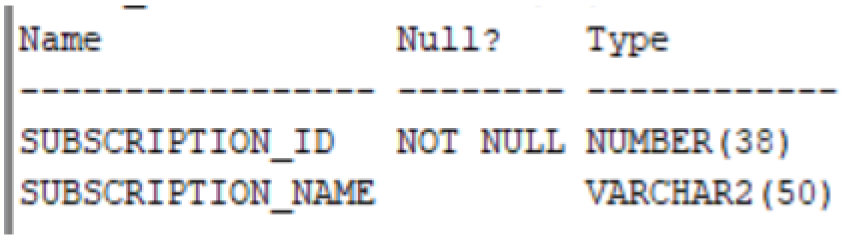
```
Name     Null?    Type
-------- -------- ----------
USER_ID  NOT NULL NUMBER(38)
GENRE_ID NOT NULL NUMBER(38)
```

SELECT * FROM PREFERRED_GENRE;



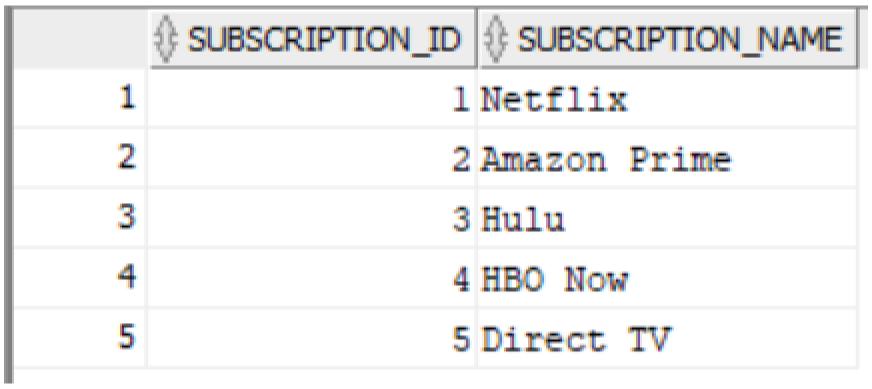
	USER_ID	GENRE_ID
1	1	100
2	2	101
3	3	102
4	4	103
5	5	104
6	6	105
7	7	106
8	8	107
9	9	108
10	10	109
11	11	110
12	12	111
13	13	112
14	14	113
15	15	114

DESCRIBE SUBSCRIPTION;



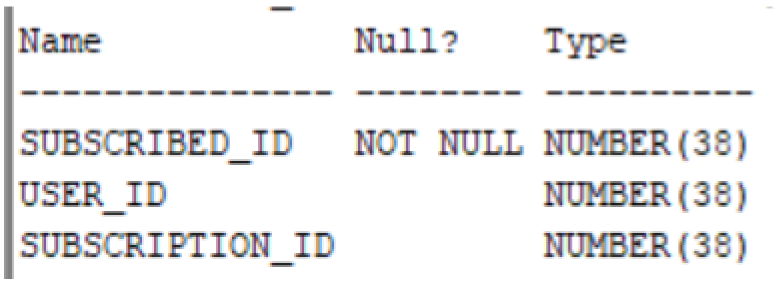
```
Name              Null?    Type
----------------- -------- ------------
SUBSCRIPTION_ID   NOT NULL NUMBER(38)
SUBSCRIPTION_NAME          VARCHAR2(50)
```

SELECT * FROM SUBSCRIPTION;



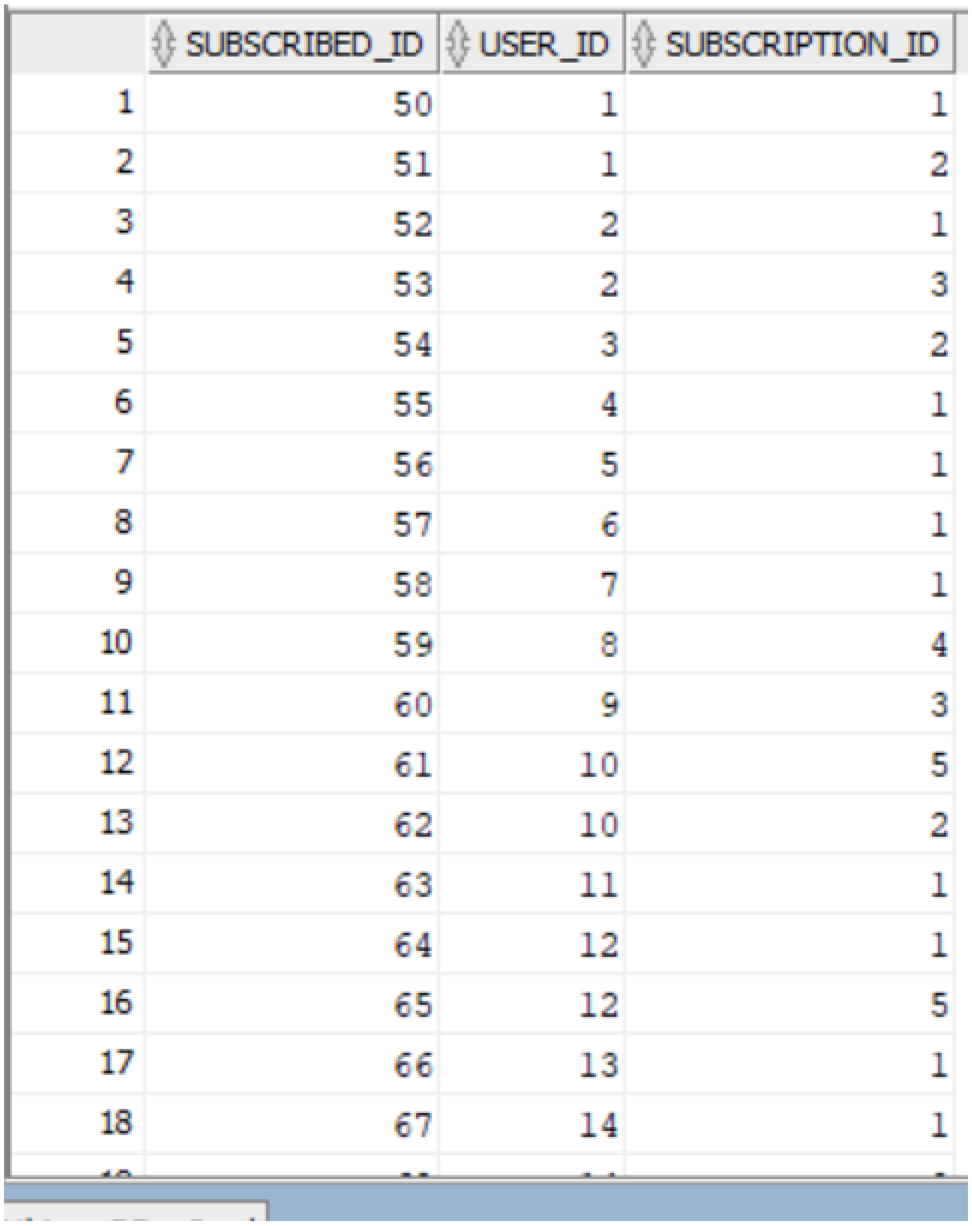
	SUBSCRIPTION_ID	SUBSCRIPTION_NAME
1	1	Netflix
2	2	Amazon Prime
3	3	Hulu
4	4	HBO Now
5	5	Direct TV

DESCRIBE SUBSCRIBED;



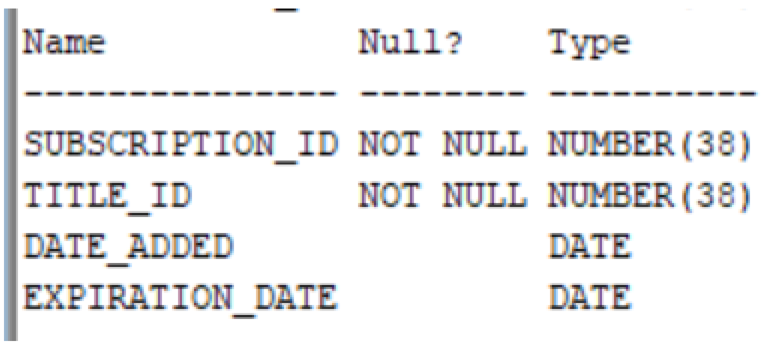
```
Name            Null?    Type
--------------- -------- ----------
SUBSCRIBED_ID   NOT NULL NUMBER(38)
USER_ID                  NUMBER(38)
SUBSCRIPTION_ID          NUMBER(38)
```

SELECT * FROM SUBSCRIBED;



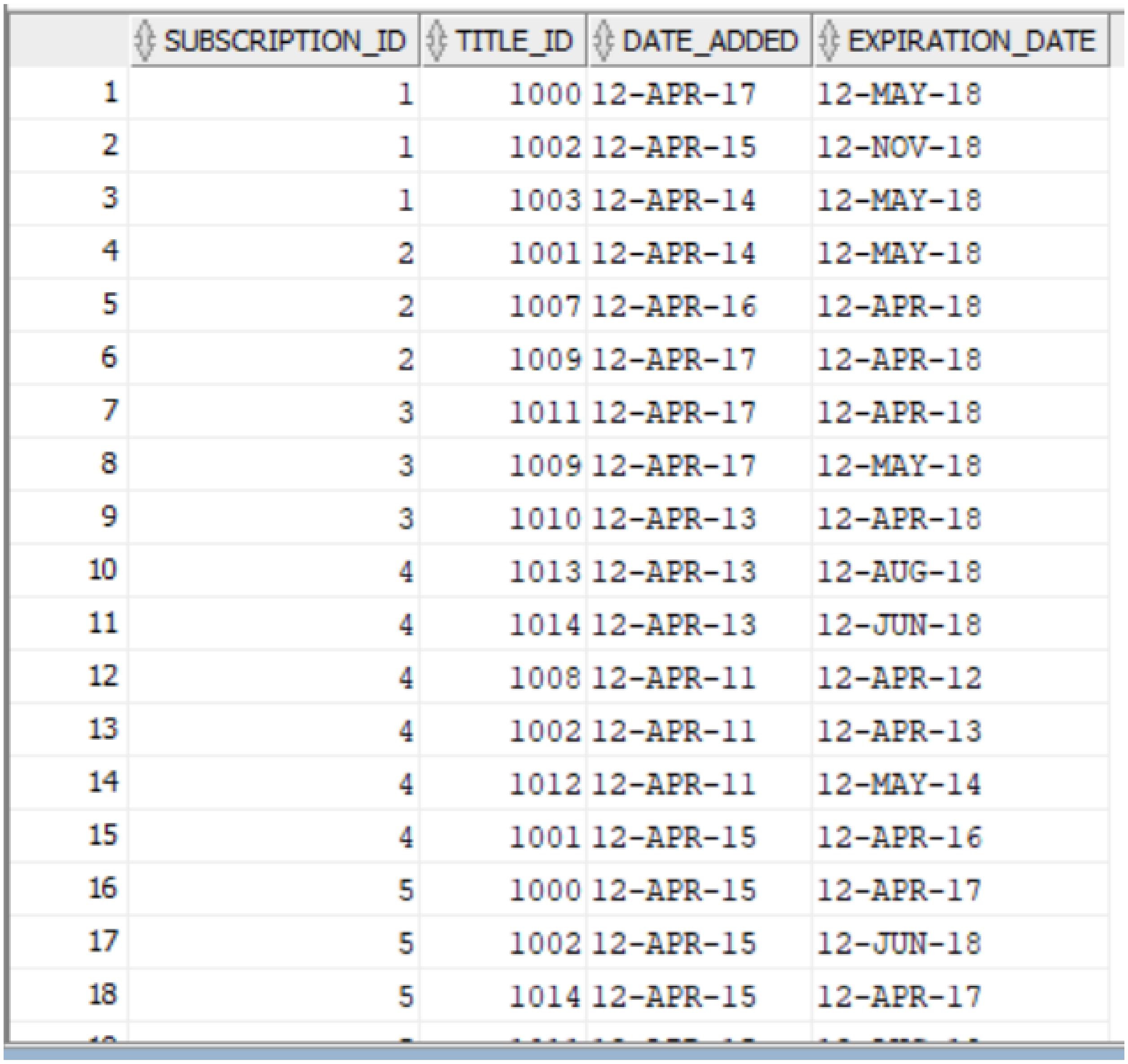
	SUBSCRIBED_ID	USER_ID	SUBSCRIPTION_ID
1	50	1	1
2	51	1	2
3	52	2	1
4	53	2	3
5	54	3	2
6	55	4	1
7	56	5	1
8	57	6	1
9	58	7	1
10	59	8	4
11	60	9	3
12	61	10	5
13	62	10	2
14	63	11	1
15	64	12	1
16	65	12	5
17	66	13	1
18	67	14	1

DESCRIBE SUBSCRIPTION_TITLE;



```
Name            Null?    Type
--------------- -------- ----------
SUBSCRIPTION_ID NOT NULL NUMBER(38)
TITLE_ID        NOT NULL NUMBER(38)
DATE_ADDED               DATE
EXPIRATION_DATE          DATE
```

SELECT * FROM SUBSCRIPTION_TITLE;



	SUBSCRIPTION_ID	TITLE_ID	DATE_ADDED	EXPIRATION_DATE
1	1	1000	12-APR-17	12-MAY-18
2	1	1002	12-APR-15	12-NOV-18
3	1	1003	12-APR-14	12-MAY-18
4	2	1001	12-APR-14	12-MAY-18
5	2	1007	12-APR-16	12-APR-18
6	2	1009	12-APR-17	12-APR-18
7	3	1011	12-APR-17	12-APR-18
8	3	1009	12-APR-17	12-MAY-18
9	3	1010	12-APR-13	12-APR-18
10	4	1013	12-APR-13	12-AUG-18
11	4	1014	12-APR-13	12-JUN-18
12	4	1008	12-APR-11	12-APR-12
13	4	1002	12-APR-11	12-APR-13
14	4	1012	12-APR-11	12-MAY-14
15	4	1001	12-APR-15	12-APR-16
16	5	1000	12-APR-15	12-APR-17
17	5	1002	12-APR-15	12-JUN-18
18	5	1014	12-APR-15	12-APR-17

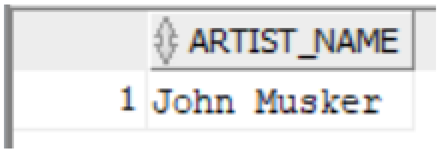
Other Table Queries

--1. Who is the most liked artist?

```
SELECT AR.ARTIST_NAME FROM (
   SELECT ARTIST_ID, COUNT(*) NO_PREFERENCE
   FROM PREFERRED_ARTIST
   GROUP BY ARTIST_ID
   ORDER BY NO_PREFERENCE DESC) PR
```

JOIN ARTIST AR ON AR.ARTIST_ID = PR.ARTIST_ID

WHERE ROWNUM = 1;



	ARTIST_NAME
1	John Musker

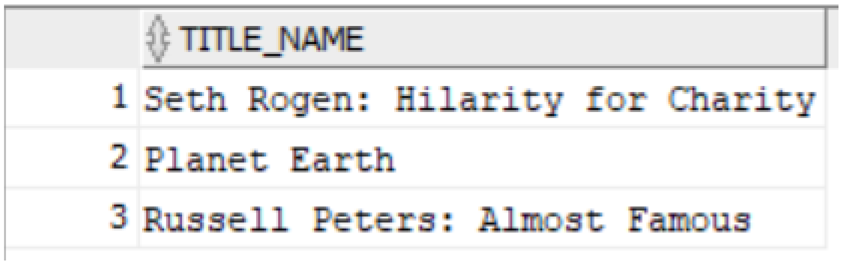
--2. List all the award winning TV shows

SELECT DISTINCT T.TITLE_NAME FROM

TITLE T JOIN

TV_SHOW TS ON T.TITLE_ID = TS.TITLE_ID JOIN

AWARD AW ON TS.TITLE_ID = AW.TITLE_ID;



	TITLE_NAME
1	Seth Rogen: Hilarity for Charity
2	Planet Earth
3	Russell Peters: Almost Famous

--3. List the titles in which the director is also the actor

SELECT TITLE_NAME FROM

TITLE T JOIN

TITLE_ROLE TR ON T.TITLE_ID = TR.TITLE_ID JOIN

ARTIST_ROLE AR ON TR.ROLE_ID = AR.ROLE_ID

WHERE AR.ROLE_NAME = 'Director'

INTERSECT

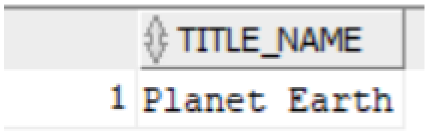
SELECT TITLE_NAME FROM

TITLE T JOIN

TITLE_ROLE TR ON T.TITLE_ID = TR.TITLE_ID JOIN

ARTIST_ROLE AR ON TR.ROLE_ID = AR.ROLE_ID

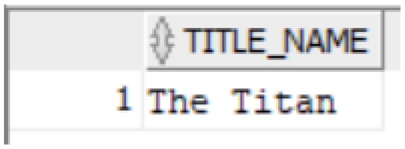
WHERE AR.ROLE_NAME = 'Actor';



```
    TITLE_NAME
1   Planet Earth
```

--4. List the movies played by John Hamburg rated above 3.5

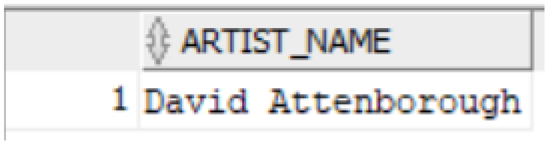
```
SELECT DISTINCT T.TITLE_NAME FROM
TITLE T JOIN
TITLE_ROLE TR ON T.TITLE_ID = TR.TITLE_ID JOIN
ARTIST AR ON TR.ARTIST_ID = AR.ARTIST_ID LEFT JOIN
TV_SHOW TS ON T.TITLE_ID = TS.TITLE_ID
WHERE T.IMDB_RATING > '3.5'
AND AR.ARTIST_NAME = 'John Hamburg'
AND TS.TITLE_ID IS NULL;
```



```
    TITLE_NAME
1   The Titan
```

-- 5. Which actor has won most awards

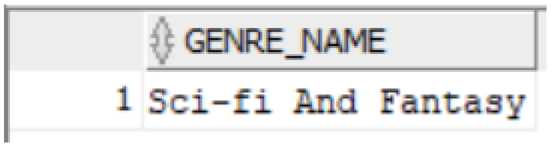
```
SELECT ART.ARTIST_NAME FROM(
  SELECT TR.ARTIST_ID, COUNT(*) AWARD_COUNT FROM
  TITLE T JOIN
  TITLE_ROLE TR ON T.TITLE_ID = TR.TITLE_ID JOIN
  ARTIST_ROLE AR ON TR.ROLE_ID = AR.ROLE_ID JOIN
  AWARD AW ON T.TITLE_ID = AW.TITLE_ID
  WHERE AR.ROLE_NAME = 'Actor'
  GROUP BY TR.ARTIST_ID
  ORDER BY AWARD_COUNT DESC) TAB1 JOIN
ARTIST ART ON TAB1.ARTIST_ID = ART.ARTIST_ID
WHERE ROWNUM = 1;
```



	ARTIST_NAME
1	David Attenborough

-- 6. Which genre receives the most awards

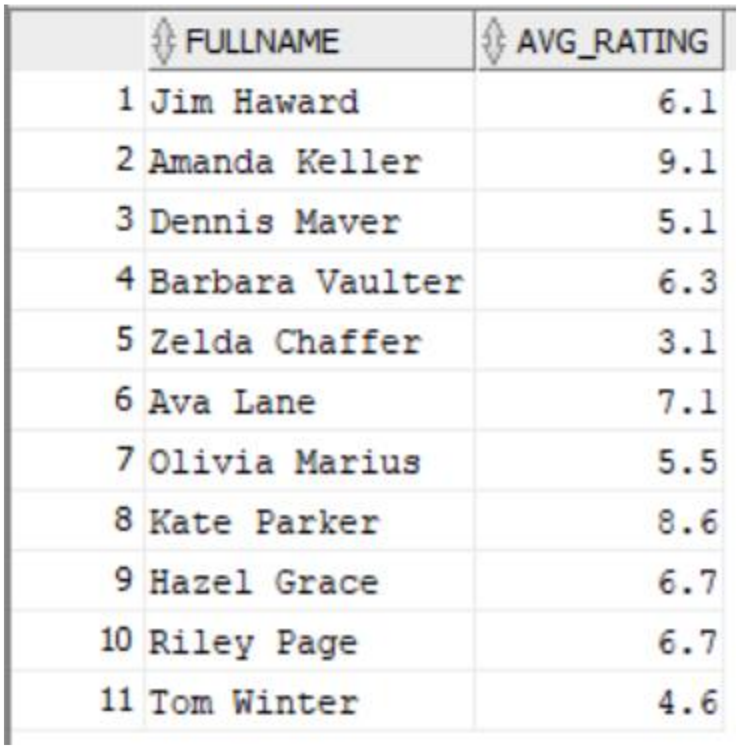
```
SELECT GENRE_NAME FROM (
   SELECT T.GENRE_ID, COUNT(*) AWARD_COUNT FROM
   TITLE_CATEGORY T JOIN
   AWARD AW ON T.TITLE_ID = AW.TITLE_ID
   GROUP BY T.GENRE_ID
   ORDER BY AWARD_COUNT DESC) TAB1 JOIN
GENRE G ON G.GENRE_ID = TAB1.GENRE_ID
WHERE ROWNUM = 1;
```



	GENRE_NAME
1	Sci-fi And Fantasy

-- 7. What is the average IMDB rating for preferred movies for each user

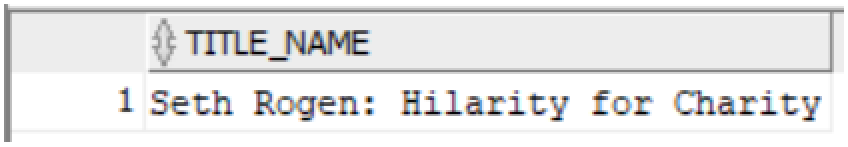
```
SELECT USER_FIRSTNAME || ' ' || USER_LASTNAME FULLNAME, TAB1.AVG_RATING FROM (
   SELECT PT.USER_ID, AVG(T.IMDB_RATING) AVG_RATING FROM
   PREFERRED_TITLE PT JOIN
   TITLE T ON T.TITLE_ID = PT.TITLE_ID LEFT JOIN
   TV_SHOW TV ON T.TITLE_ID = TV.TITLE_ID
   WHERE TV.TITLE_ID IS NULL
   GROUP BY PT.USER_ID) TAB1 JOIN
APP_USER AU ON TAB1.USER_ID = AU.USER_ID;
```



	FULLNAME	AVG_RATING
1	Jim Haward	6.1
2	Amanda Keller	9.1
3	Dennis Maver	5.1
4	Barbara Vaulter	6.3
5	Zelda Chaffer	3.1
6	Ava Lane	7.1
7	Olivia Marius	5.5
8	Kate Parker	8.6
9	Hazel Grace	6.7
10	Riley Page	6.7
11	Tom Winter	4.6

-- 8. Which is the highest rated award winning TV Show since 1990

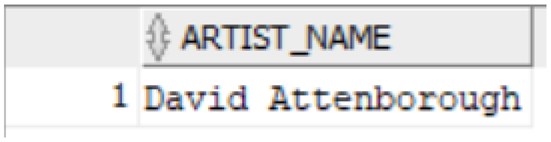
```
SELECT TI.TITLE_NAME FROM (
  SELECT T.TITLE_ID, IMDB_RATING FROM
  AWARD AW JOIN
  TV_SHOW TV ON AW.TITLE_ID = AW.TITLE_ID JOIN
  TITLE T ON TV.TITLE_ID = T.TITLE_ID
  ORDER BY IMDB_RATING
  ) TAB1 JOIN
TITLE TI ON TI.TITLE_ID = TAB1.TITLE_ID
WHERE ROWNUM = 1;
```



	TITLE_NAME
1	Seth Rogen: Hilarity for Charity

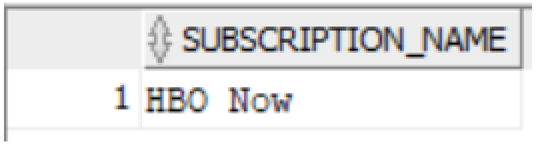
-- 9. Pick the actor that has worked in most genres

```
SELECT ARTIST_NAME FROM (
  SELECT ARTIST_ID, COUNT(*) GENRE_COUNT FROM (
    SELECT DISTINCT TR.ARTIST_ID, TC.GENRE_ID FROM
    TITLE_CATEGORY TC JOIN
    TITLE_ROLE TR ON TC.TITLE_ID = TR.TITLE_ID) CAT
    GROUP BY ARTIST_ID
    ORDER BY GENRE_COUNT DESC
  ) TAB1 JOIN
ARTIST AR ON TAB1.ARTIST_ID = AR.ARTIST_ID
WHERE ROWNUM = 1;
```



	ARTIST_NAME
1	David Attenborough

```
-- 10. Which is the most subscribed app
SELECT SUB.SUBSCRIPTION_NAME FROM (
  SELECT SU.SUBSCRIPTION_ID, COUNT(*) SUB_COUNT FROM
  SUBSCRIBED SU
  GROUP BY SU.SUBSCRIPTION_ID
  ORDER BY SUB_COUNT
  ) TAB1 JOIN
SUBSCRIPTION SUB ON TAB1.SUBSCRIPTION_ID = SUB.SUBSCRIPTION_ID
WHERE ROWNUM = 1;
```



	SUBSCRIPTION_NAME
1	HBO Now

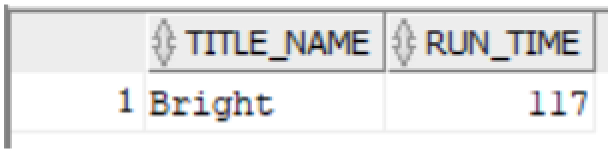
```
-- 11. Which movie has the longest runtime
SELECT TITLE_NAME, RUN_TIME FROM
```

TITLE

WHERE RUN_TIME =

 (SELECT MAX(RUN_TIME) FROM

 TITLE);



	TITLE_NAME	RUN_TIME
1	Bright	117

-- 12. List the titles in Netflix

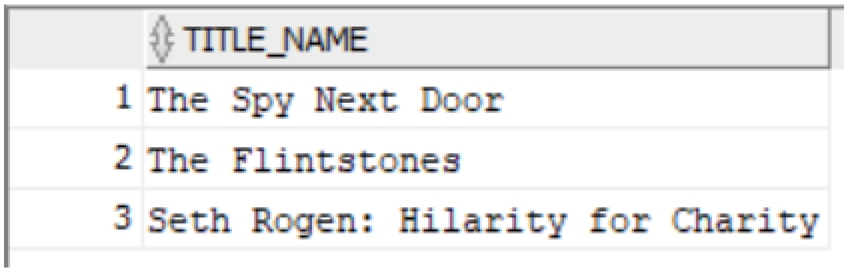
SELECT T.TITLE_NAME FROM

TITLE T JOIN

SUBSCRIPTION_TITLE ST ON T.TITLE_ID = ST.TITLE_ID JOIN

SUBSCRIPTION SUB ON ST.SUBSCRIPTION_ID = SUB.SUBSCRIPTION_ID

WHERE SUB.SUBSCRIPTION_NAME = 'Netflix';



	TITLE_NAME
1	The Spy Next Door
2	The Flintstones
3	Seth Rogen: Hilarity for Charity

-- 13. List titles that exist in more than one subscription

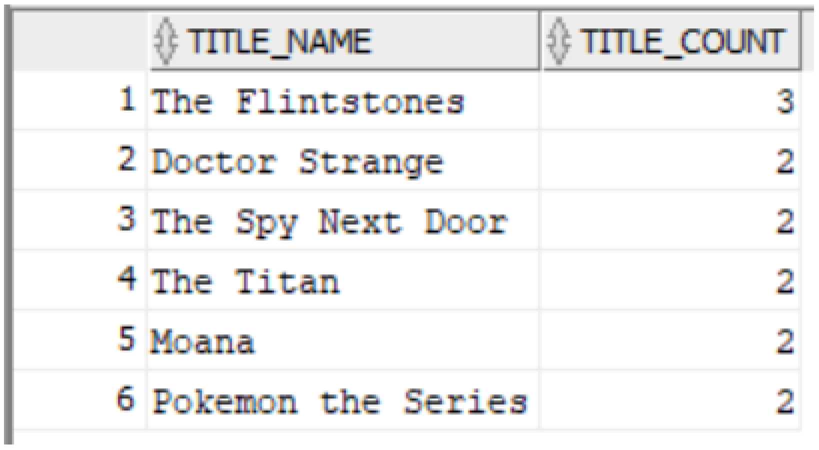
SELECT T.TITLE_NAME, COUNT(*) TITLE_COUNT FROM

SUBSCRIPTION_TITLE ST JOIN

TITLE T ON T.TITLE_ID = ST.TITLE_ID

GROUP BY T.TITLE_NAME

HAVING COUNT(*) > 1;



	TITLE_NAME	TITLE_COUNT
1	The Flintstones	3
2	Doctor Strange	2
3	The Spy Next Door	2
4	The Titan	2
5	Moana	2
6	Pokemon the Series	2

-- 14. Which is the most preferred genre

SELECT GENRE_NAME FROM (

SELECT GENRE_ID, COUNT(*) PREFERRED_COUNT FROM

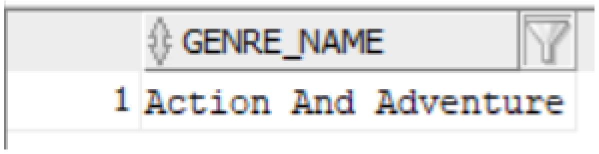
PREFERRED_GENRE PG

GROUP BY GENRE_ID

ORDER BY PREFERRED_COUNT DESC) TAB1

JOIN GENRE G ON G.GENRE_ID = TAB1.GENRE_ID

WHERE ROWNUM = 1;



	GENRE_NAME
1	Action And Adventure

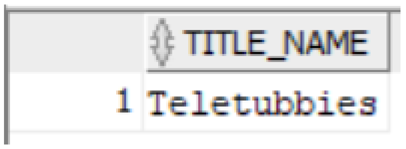
-- 15. List all the titles that toddlers are allowed to watch.

SELECT T.TITLE_NAME FROM

TITLE T JOIN

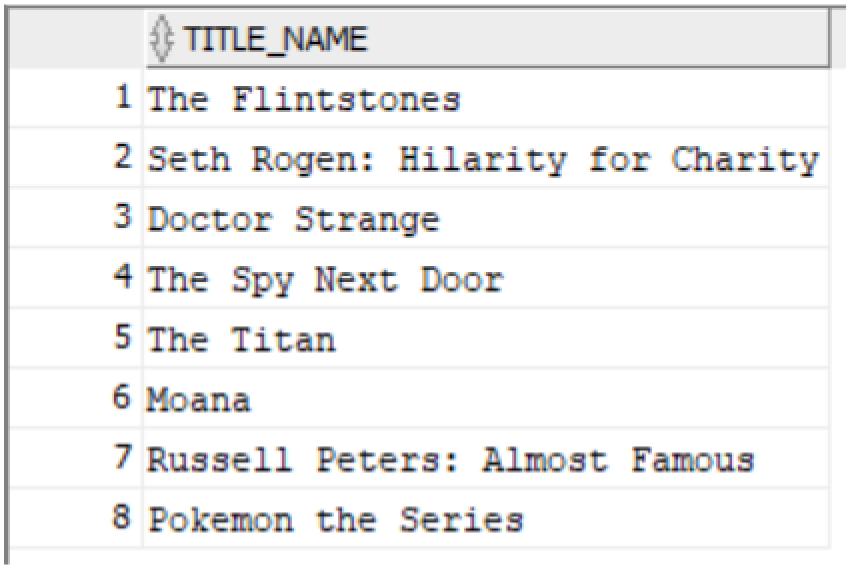
TV_RATING TR ON T.TV_RATING_ID = TR.TV_RATING_ID

WHERE TR.RATING_NAME = 'TV-Y';



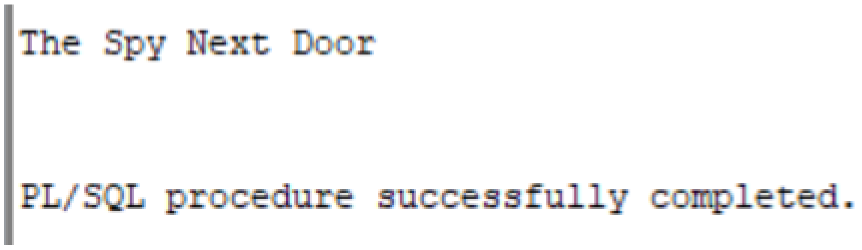
	TITLE_NAME
1	Teletubbies

```
-- 16. ist the titles that are yet to be expired amongst all subscriptions

SELECT DISTINCT T.TITLE_NAME FROM

   SUBSCRIPTION_TITLE ST JOIN

   TITLE T ON T.TITLE_ID = ST.TITLE_ID

   WHERE ST.EXPIRATION_DATE > SYSTIMESTAMP;
```



	TITLE_NAME
1	The Flintstones
2	Seth Rogen: Hilarity for Charity
3	Doctor Strange
4	The Spy Next Door
5	The Titan
6	Moana
7	Russell Peters: Almost Famous
8	Pokemon the Series

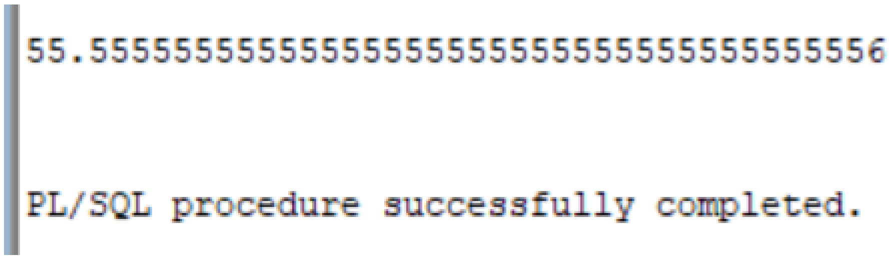
```
--------------------------------------------------------------------------------------------------------------------

--                                        ** PL/SQL QUERYING **


SET SERVEROUTPUT ON;


-- 1. Print the next title to expire amongst all subscriptions

DECLARE

CURR_TIME TIMESTAMP;

NEXT_TITLE VARCHAR(500);

BEGIN

   CURR_TIME := SYSTIMESTAMP;
   SELECT T.TITLE_NAME INTO NEXT_TITLE FROM
   SUBSCRIPTION_TITLE ST JOIN
   TITLE T ON T.TITLE_ID = ST.TITLE_ID
   WHERE ST.EXPIRATION_DATE  = (SELECT MIN(EXPIRATION_DATE)
                           FROM SUBSCRIPTION_TITLE
                           WHERE EXPIRATION_DATE > CURR_TIME)
                           AND ROWNUM = 1;
   DBMS_OUTPUT.PUT_LINE(NEXT_TITLE);
END;
```



```
The Spy Next Door


PL/SQL procedure successfully completed.
```

```
-- 2. What percentage of titles have won awards after being nominated
DECLARE
   TOTAL INTEGER;
   WON_NUMBER INTEGER;
BEGIN
   SELECT  COUNT(*) INTO TOTAL FROM (
      SELECT DISTINCT TITLE_ID FROM
      AWARD) AW;

   SELECT  COUNT(*) INTO WON_NUMBER FROM (
      SELECT DISTINCT TITLE_ID FROM
      AWARD WHERE WON = 'Yes') AW;

   DBMS_OUTPUT.PUT_LINE( WON_NUMBER/TOTAL * 100);
```

END;



```
55.55555555555555555555555555555555555556


PL/SQL procedure successfully completed.
```