CSCE 5350.001 Fundamentals of Database Systems Project Part 2

Naga Vara Pradeep Yendluri 11646461 nagavarapradeepyendluri@my.unt.edu

Project Description:

The Movie Producer Management System is an application that is being developed for a movie production company like Universal Studios. The system is designed to store and manage information about the company's movies, artists, songs, employees and various other aspects of the movie production process. The system will store information about the producing site locations, movie-script-inventory, sponsoring companies, employee data, and payroll. It will also store information about the artists and the movies they have worked on, as well as the various aspects of the movie production process, such as soundtracks, awards, and more.

We have Identified the following entities and relations for the movie producer management system.

- 1. **Movies:** The entity 'Movies' provides information about the different movies produced by the company. It has 5 attributes, including the movie id, movie title, release date, duration and script inventory id. This entity is important for keeping track of the different movies produced by the company and the information related to each movie.
- 2. Artists: The entity 'Artists' provides information about the actors involved in the movies. It has 4 attributes, including the artist id, artist name, artist date of birth and gender. This entity is important for maintaining the information about the actors, their age and date of birth, which is required for casting actors for various roles.
- 3. **Genre:** The entity 'Genre' provides information about the genre to which the movie belongs. It has 2 attributes, genre id, and genre name. This entity is important for categorizing the movies into different genres, which helps in better management and analysis of the movies.
- 4. **Sponsoring Companies:** The entity 'Sponsoring Companies' provides information about the companies that sponsor the movies. It has 2 attributes, including company id and company name. This entity is important for tracking the sponsorship deals and the companies that sponsor the movies.
- 5. **Site Locations:** The entity 'Site Locations' provides information about the different producing sites, including their addresses and buildings. It has 3 attributes, including location id, name, and address. This entity is important for tracking the different producing sites and their details, which is essential for managing the movie production process.
- 6. **Buildings:** The entity 'Buildings' provides information about the buildings in each producing site. It has 4 attributes, including Building id, name, location id and purpose.

This entity is important for tracking the different types of buildings present in the producing sites, which is essential for managing the resources and maintenance of the buildings.

- 7. **Movie Script Inventory:** The entity 'Movie Script Inventory' provides information about the movie scripts. It has 2 attributes, including script inventory id and script inventory name. This entity is important for tracking the different movie scripts and the information related to each script.
- 8. **Employees:** The entity 'Employees' provides information about the employees of the company. It has 4 attributes, including employe id, name, designation, and phone. This entity is important for maintaining the information about the employees, their job title and contact information, which is required for managing the human resources of the company.
- 9. **Payroll:** The entity 'Payroll' provides information about employee payroll data. It has 4 attributes, including payroll id, salary, employee id and hours worked. This entity is important for tracking the payroll information of the employees, which is essential for managing the finances of the company.
- 10. **Songs:** The entity 'Songs' provides information about different soundtracks used in the movies. It has 4 attributes, including song id, song name, movie id, and singer name. This entity is important for tracking the different soundtracks used in the movies and the information related to each soundtrack.

Binary Relations:

• One-to-Many relation:

- 1. Every Employee will have one payroll associated to them each month. So, the relation between employee and payroll is one to many.
- 2. Each script inventory will have many movies associated with. Hence, the relation between movie script inventory and movies will also be one to many.
- 3. Each movie can have multiple songs/soundtracks in it and inversely there can be multiple songs in a movie. So, the relation between songs and movies is one to many.
- 4. Each site location can have multiple buildings in them, and inversely multiple buildings can be located at a single location. So, the relation between these two entities will be one-to-many relation.

• Many-to-Many relation:

- 1. Each movie can have many artists performing in it and inversely many actors can act in many different movies. Hence, the relation between movies and artists will be many to many.
- 2. Many Sponsoring companies can sponsor for many movies in this system. Inversely, many movies can get sponsorship from different sponsoring companies. Hence, the relation between these two entities is many to many relation.
- 3. Many movies can be shot at different locations and inversely many locations can concurrently host shootings for many movies in different buildings so the relation between these two entities will also be many to many relation.
- 4. Many movies can have many genres so the relation between these two entities will also be many to many relation.

- 5. Many employees can manage many site locations so the relation between these two entities will also be many to many relation.
- 6. Many artists gets paid by many sponsoring companies so the relation between these two entities will also be many to many relation.

Assumptions:

- 1. Movies and Artists: It is assumed that each movie has one or more actors, and each actor is associated with one or more movies.
- 2. Genre: It is assumed that each movie belongs to multiple genres like adventure, action, drama, etc.
- 3. Sponsoring Companies: It is assumed that each movie is sponsored by one or more companies and each company sponsors one or more movies.
- 4. Site Locations and Buildings: It is assumed that each producing site has one or more buildings and each building is located at a single site.
- 5. Movie Script Inventory: It is assumed that each movie is associated with one unique movie scripts and each script is associated with a single movie.
- 6. Employee: It is assumed that each employee of the company has a unique employee ID, and each employee belongs to a single job title.
- 7. Payroll: It is assumed that each employee has payroll information, and each employee ID is associated with a single payroll record.
- 8. Songs: It is assumed that each movie has one or more soundtracks, and each soundtrack is associated with a single movie.
- 9. Movie title must be unique.
- 10. Ratings must be between 1 to 10 and no decimals values.
- 11. Employees get paid by weekly basis hence we used 40 hours constraint in the table.
- 12. Company name must be unique.
- 13. Phone number has 10 characters.
- 14. Employee will have unique phone number.
- 15. One employee can manage multiple locations. m-m
- 16. Duration of movie must be > 0 minutes.
- 17. Artist gender must be of F(female), M(male), T(transgender).

<u>ER Diagram</u>



ER Relations:

Relations transformed to schema.

Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id)

Songs (song_id, song_name, singer_name, movie_id)

Genre (genre_id, genre_name)

TaggedWith (movie_id, genre_id)

SiteLocation (location_id, location_name, address)

ShotAt (location_id, movie_id)

Building (building_id, building_name, purpose, location_id)

PostProductionDoneIn (movie_id, building_id)

Employees (employee_id, employee_name, designation, phone_number)

Manages (employee_id, location_id)

Payroll (**payroll_id**, salary, employee_id, hours_worked)

SponsoringCompany (company_id, company_name)

getsPaidBy (artist_id, company_id)

Produces (company_id, movie_id)

Artist (artist_id, artist_name, date_of_birth, gender)

MovieScriptInventory (script_inventory_id, script_inventory_name)

ActsIn(**movie_id**, artist_id)

Tables:

MovieScriptInventory:

CREATE TABLE MovieScriptInventory (script_id INT PRIMARY KEY, script_name VARCHAR(255) NOT NULL);

SqL Plus × + ✓	-	X
SQL*Plus: Release 21.0.0.0.0 - Production on Sun Mar 5 17:08:17 2023 Version 21.3.0.0.0		
Copyright (c) 1982, 2021, Oracle. All rights reserved.		
Enter user-name: system Enter password: Last Successful login time: Sun Mar 05 2023 17:07:11 -06:00		
Connected to: Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production Version 21.3.0.0.0		
SQL> DROP TABLE MovieScriptInventory CASCADE CONSTRAINTS;		
Table dropped.		
<pre>SQL> rem ++ SQL> rem + Create MovieScriptInventory Table SQL> rem ++ SQL> SQL> CREATE TABLE MovieScriptInventory (2 script_inventory_id INT PRIMARY KEY, 3 script_inventory_name VARCHAR(255) NOT NULL 4);</pre>		
Table created.		
sqL>		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (1, 'The Shawshank Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (2, 'The Avengers Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (3, 'The Dark Knight Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (4, 'Fiction Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (5, 'The Lord of the Rings Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (6, 'Forrest Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (7, 'The Silence Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (8, ' Matrix Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (9, 'Ryan Inventory');		
l row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (10, 'Inception Inventory');		
1 row created.		
SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (9, 'Ryan Inventory');		
<pre>1 row created. SQL> INSERT INTO MovieScriptInventory (script_inventory_id, script_inventory_name) 2 VALUES (10, 'Inception Inventory');</pre>		
1 row created.		

E 301 Plus X + ↓ ↓		o	×
SQL> select * from MovieScriptInventory;			
SCRIPT_INVENTORY_ID			
SCRIPT_INVENTORY_NAME			
1 The Shawshank Inventory			
2 The Avengers Inventory			
3 The Dark Knight Inventory			
SCRIPT_INVENTORY_ID			_
SCRIPT_INVENTORY_NAME			
4 Fiction Inventory			_
5 The Lord of the Rings Inventory			
6 Forrest Inventory			
SCRIPT_INVENTORY_ID			
SCRIPT_INVENTORY_NAME			
7			
E SQL Plus X + v	-	0	×
4 Fiction Inventory			
5 The Lord of the Rings Inventory			
6 Forrest Inventory			
SCRIPT_INVENTORY_ID			
SCRIPT_INVENTORY_NAME			
7 The Silence Inventory			
8 Matrix Inventory			
9 Ryan Inventory			

Movie:

CREATE TABLE Movie (movie_id INT NOT NULL, title VARCHAR(255) NOT NULL UNIQUE, rating INT NOT NULL, date_of_release DATE NOT NULL, duration INT NOT NULL, script_inventory_id INT NOT NULL, PRIMARY KEY (movie_id), FOREIGN KEY (script_inventory_id) REFERENCES MovieScriptInventory(script_inventory_id), CONSTRAINT rating_constraint_violated CHECK (rating > 0 and rating < 11), CONSTRAINT duration_constraint_violated CHECK (duration > 0));

SQL> DROP TABLE Movie CASCADE CONSTRAINTS;
Table dropped.
SQL> rem +
<pre>SQL> SQL>CEATE TABLE Movie (2 movie_id INT NOT NULL, 3 title VARCHAR(255) NOT NULL UNIQUE, 4 rating INT NOT NULL, 5 date_of_release DATE NOT NULL, 6 duration INT NOT NULL, 7 script_inventory_id INT NOT NULL, 8 PRIMARY KEY (movie_id), 9 FOREGM KEY (movie_id), 10 CONSTRAINT rating_constraint_violated CHECK (11 rating > 0 12 and rating < 11 13), 14 CONSTRAINT duration_constraint_violated CHECK (duration > 0) 15);</pre>
Table created. SOL>
S SQLPlus X + ↓ − O X Table created
SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (1, 'The Godfather', 9, TO_DATE('1972-03-24', 'yyyy-mm=dd'), 175, 1); 1 mmm created
SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (2, 'The Shamshank Redemption', 9, TO_DATE('1994-09-23', 'yyyy=mm=dd'), 142, 2);
<pre>SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (3, 'The Dark Knight', 9, TO_DATE('2008-07-18', 'yyyy-mm=dd'), 152, 3);</pre>
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (4, 'The Godfather: Part II', 9, TO_DATE('1974-12-28', 'yyyy=mm=dd'), 202, 1);
l row created. SQL> INSERT INTO Movie (movie.id, title, rating, date.of_release, duration, script_inventory_id) 2 VALUES (S, 'The Lord of the Rings: The Return of the King', 9, TO_DATE('2003-12-17', 'yyyy=mm=dd'), 201, 4);
l row created. SQL> INSERT INTO Movie (movie.id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (6, 'Pulp Fiction', 9, TO_DATE('1994-10-14', 'yyyy-mm-dd'), 154, 5);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (7, 'Schindler''s List', 8, TO_DATE('1993-12-15', 'yyyy-mm-dd'), 195, 6);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (8, 'The Lord of the Rings: The Fellowship of the Ring', 8, TO_DATE('2001-12-19', 'yyyy-mm-dd'), 178, 4);
l row created. SQL> TNSEET INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (9, 'The Lord of the Rings: The Two Towers', 8, TO_DATE('2002-12-18', 'yyyy-mm-dd'), 179, 4); 1 row created
SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id)
図 SQL Plus × + の ×
SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (10, 'Forrest Gump', 8, TO_DATE('1994-07-06', 'yyyy-mm-dd'), 142, 7);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (11, 'Inception', 8, T0_DATE('2018-07-16', 'yyyy=mm=dd'), 148, 8);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (12, 'Fight Club', 8, T0_DATE('1999-10-15', 'yyyy-mm-dd'), 139, 9);
1 row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (13, 'The Matrix', 8, TO_DATE('1999-03-31', 'yyyy-mm-dd'), 136, 10);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (14, 'Goodfellas', 8, TO_DATE('1990-89-19', 'yyyy-mm-dd'), 146, 9);
1 row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (15, 'The Silence of the Lambs', 8, TO_DATE('1991-82-14', 'yyyy-mm-dd'), 118, 9);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (16, 'The Departed', 8, TO_DATE('2006-10-06', 'yyyy-mm=-dd'), 151, 8);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (17, 'Saving Private Ryan', 8, TO_DATE('1998-07-24', 'yyyy-mm-dd'), 169, 9);
l row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (18, 'Terminator 2: Judgment Day', 8, TO_DATE('1991-07-03', 'yyyy-mm-dd'), 137, 6);
1 row created. SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (19, 'The Prestige', 8, TO_DATE('2006-10-20', 'yyyy=mm=dd'), 130, 8);
SQL> INSERT INTO Movie (movie_id, title, rating, date_of_release, duration, script_inventory_id) 2 VALUES (20, 'Gladiator', 8, TO_DATE('2008-05-01', 'yyyy-mm-dd'), 155, 6); 1 row created.

SQL Plus			
SQL> select * from Movi			
MOVIE_ID			
TITLE			
RATING DATE_OF_R	DURATION SCRIPT_INVENTORY_	ID	
1 The Godfather 9 24-MAR-72	175		
2 The Shawshank Redemptio 9 23-SEP-94	in 142		
MOVIE_ID			
TITLE			
RATING DATE_OF_R	DURATION SCRIPT_INVENTORY_	_TD	
3 The Dark Knight 9 18-JUL-08	152		
4 The Godfather: Part II			
MOVIE_ID			
TITLE			

SQL Plus	× + ~	
 TITLE		
RATING DATE_OF_R		
9 20-DEC-74	202	
5 The Lord of the Rings 9 17-DEC-03	: The Return of the King 201	
6		
MOVIE_ID		
TITLE		
RATING DATE_OF_R	DURATION SCRIPT_INVENTOR	Y_ID
Pulp Fiction		
9 14-0CT-94	154	
7 Schindler's List		
8 15-DEC-93	195	
MOVIE TO		
TITLE		
RATING DATE_OF_R	DURATION SCRIPT_INVENTOR	Y_ID
8 The Lord of the Rings	: The Fellowship of the Ri	ng

SQL Plus					
8	19-DEC-01	178	4		
9 The Lord o 8	f the Rings: 18-DEC-02	The Two T 179	Towers 4		
MOVIE_ID					
TITLE					
RATING	DATE_OF_R	DURATION	SCRIPT_INVENTORY_ID		
10 Forrest Gu 8	mp 06-JUL-94	142	7		
11 Inception					
MOVIE_ID					
TITLE					
RATING	DATE_OF_R	DURATION	SCRIPT_INVENTORY_ID	D	
8	16-JUL-10	148	8	8	
12 Fight Club 8	15-0CT-99	139	9		
13					
MOVIE_ID					

S SOL Plus X + V		٥	
RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID			
The Matrix			
8 31-HAR-99 136 10			
14 Gondfellas			
8 19-SEP-90 146 9			
MOVIE TO			
PATTER			
The Silence of the Lambs			
0 19776591 110 9			
The Departed			
RU14_1D			
111tt			
17			
E squ Mus × + v	-	0	×
Saving Private Ryan 8 24-JUL-98 169 9			
Saving Private Byan 8 24-JUL-98 169 9 18			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID 			
Saving Private Ryan 8 2H-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID 			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 19 The Prestige			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 26-OCT-06 130 8			
Saving Private Ryan 8 24-JU-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 20-OCT-06 130 8 20			
Saving Private Ryan 8 24-JUL-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 20-OCT-06 130 8 20 MOVIE_ID			
Saving Private Ryan 8 241-0U-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 20-OCT-06 130 8 20 MOVIE_ID TITLE			
Saving Private Ryan 8 241-0U-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 20-OCT-06 130 8 20 MOVIE_ID TITLE TRATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID TABLE D			
Saving Private Ryan 8 24-JU-98 169 9 18 Terminator 2: Judgment Day MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 8 03-JUL-91 137 6 The Prestige 8 20-OCT-06 130 8 20 MOVIE_ID TITLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID Gladiator 0 15 6			
Saving Private Ryan 18 Terminator 2: Judgment Day MOVIE_ID THILE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID 19 The Prestige 8 03-JUL-91 137 6 19 The Prestige 8 03-JUL-91 137 6 20 MOVIE_ID THILE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID Cladiator 8 01-MAY-00 155 6			
Saving Private Ryan 18 Terminator 2: Judgment Day MOVIE_ID THE 19 The Prestige 8 03-JUL-91 137 6 19 The Prestige 20 MOVIE_ID THLE THELE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID CONTE_ID THLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID Cladiator 8 01-MAY-00 155 6 20 rows selected.			
Saving Private Ryan 324-JU-98 169 9 Terminator 2: Judgment Day MOVIE_ID THE 19 The Prestige 3 09-JU-91 137 6 19 The Prestige 3 09-JU-91 137 6 20 MOVIE_ID THLE THLE RATING DATE_OF_R DURATION SCRIPT_INVENTORY_ID Gladiator 8 01-MAY-09 155 6 20 rows selected. SQL>			

Songs:

CREATE TABLE Songs (song_id INT NOT NULL, song_name VARCHAR(255) NOT NULL, singer_name VARCHAR(255) NOT NULL, movie_id INT NOT NULL, PRIMARY KEY (song_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id));

SQL> DRDP TABLE Songs CASCADE CONSTRAINTS;
Table dropped.
SQL> ren +
\$0.> rea i
SQL> CREATE TABLE Songs (2 song id INT NOT NULL,
3 song_name_VARCHAR(255) NOT MULL, 4 singer_name_VARCHAR(255) NOT MULL
5 movie_ad_INT_NOT_NULL, 6 pertumber Ver (const id)
7 FOREIGN KEY (movie_id) REFERENCES Movie_id)

SQL Plus				
Table created.				
SQL> INSERT INTO Son 2 VALUES (1, 'Sha	gs (song_id, song_name, singer_name pe of You', 'Ed Sheeran', 1);	, movie_id)		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (2, 'Bil	gs (song_id, song_name, singer_name Lie Jean', 'Michael Jackson', 2);	, movie_id)		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (3, 'Boh	gs (song_id, song_name, singer_name emian Rhapsody', 'Queen', 3);	, movie_id)		
l row created.				
SQL> INSERT INTO Son 2 VALUES (4, 'Sta	gs (song_id, song_name, singer_name irway to Heaven', 'Led Zeppelin', 4	, movie_id));		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (5, 'Swe	gs (song_id, song_name, singer_name et Child O Mine', 'Guns N Roses', 5	, movie_id));		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (6, 'Sme	gs (song_id, song_name, singer_name lls Like Teen Spirit', 'Nirvana', 6	, movie_id));		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (7, 'Hot	gs (song_id, song_name, singer_name el California', 'Eagles', 7);	, movie_id)		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (8, 'Don	gs (song_id, song_name, singer_name t Stop Believin', 'Journey', 8);	, movie_id)		
1 row created.				
SQL> INSERT INTO Son 2 VALUES (9, 'Thr	gs (song_id, song_name, singer_name iller', 'Michael Jackson', 9);	, movie_id)		
1 row created.				

SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (10, 'Wonderwall', 'Oasis', 10);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (11, 'Uptown Funk', 'Mark Ronson ≠t. Bruno Mars', 1);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (12, 'Smooth', 'Santana ft. Rob Thomas', 2);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (13, 'Livin on a Prayer', 'Bon Jovi', 3);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (14, 'Hello', 'Adele', 4);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (15, 'I Will Always Love You', 'Whitney Houston', 5);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (16, 'Every Breath You Take', 'The Police', 6);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (17, 'Eye of the Tiger', 'Survivor', 7);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (18, 'Nothing Else Matters', 'Metallica', 8);
1 row created.
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id) 2 VALUES (19, 'Let It Be', 'The Beatles', 9);
SQL> INSERT INTO Songs (song_id, song_name, singer_name, movie_id)

Ø

OUTPUT:

1 row created.

S 30L Plus × + √	-	o	×
SQL> select * from Songs;			
SONG_ID			
SONG_NAME			
SINGER_NAME			
MOVIE_ID			
I Shape of You Ed Sheeran 1			
SONG_ID			
SONG_NAME			
SINGER_NAME			
ROVIE_ID			
Bittle Jean Michael Jackson 2			
SONG_ID			
SONG_NAME			
SINGER_NAME			

SQL Plus	\times + \sim	- 0	×
MOVIE_ID			
3			
Bohemian Rhapsody Oueen			
3			
SONG_ID			
SONG_NAME			
SINGER_NAME			
MOVIE_ID			
4			
Led Zeppelin	en de la companya de		
4			
SONG_ID			
SONG_NAME			
SINGER_NAME			
MOVIE_ID			
5 Sweet Child O Min	1e		
Guns N Roses			

5 SONG_ID SONG_NAME MOVIE_ID 6 Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME Song_NAME MovIE_ID 7 Song_NAME 1 Song_NAME	SQL Plus	× + ~	-	×
SONG_ID SONG_NAME SINGER_NAME MOVIE_ID Song_ID SONG_ID SONG_NAME SINGER_NAME MOVIE_ID 7 Hotel California Eagles 7				
SONG_ID SONG_NAME SINGER_NAME MOVIE_ID SONG_ID SONG_NAME SINGER_NAME SONG_ID SONG_NAME SINGER_NAME MOVIE_ID TO SONG_ID SONG_NAME SINGER_NAME MOVIE_ID TO SONG_NAME MOVIE_ID TO SONG_NAME MOVIE_ID TO SONG_NAME SONG_NAME MOVIE_ID TO SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME SONG_NAME				
SONG_NAME SINGER_NAME SINGER_NAME SINGER_NAME SonG_ID SONG_NAME SINGER_NAME SINGER_NAME SINGER_NAME SONG_ID SONG_NAME SINGER_NAME SINGER_NAME SONG_NAME	SONG_ID			
SINGER_NAME MOVIE_ID 	SONG_NAME			
MOVIE_ID 	SINGER_NAME			
6 Seells Like Teen Spirit Nirvana 6 SonG_ID SonG_NAME SINGER_NAME MovIE_ID T Hotel California Eagles 7 SonG_ID SonG_ID SonG_ID SonG_ID SonG_ID	MOVIE_ID			
Song_ID Song_ID Song_NAME SINGE_NAME MOVIE_ID T Hotel California Eagles 7 Song_ID Song_NAME () () () () () () () () () ()	6			
6 SONG_ID SONG_NAME SINGER_NAME MOVIE_ID Hotel California Eagles 7 SONG_ID SONG_ID SONG_NAME	Smells Like Teen Spirit Nirvana			
SONG_ID SONG_NAME SINGER_NAME MOVIE_ID 7 Hotel California Eagles 7 SONG_ID 50NG_ID SONG_ID				
SONG_ID SONG_NAME SINGE_NAME MOVIE_ID T Hotel California Eagles 7 SONG_ID SONG_NAME	60NG TD			
SONG_NAME SINGER_NAME MOVIE_ID T Hotel california Eagles 7 SONG_ID SONG_NAME	SONG_ID			
SINGER_NAME	SONG_NAME			
MOVIE_ID 7 Hotel California Eagles 7 SONG_ID SONG_NAME	SINGER_NAME			
To the california Eagles 7 SONG_ID SONG_NAME	MOVIE_ID			
Hotel California Eagles 7 SONG_ID SONG_NAME	7			
7 SONG_ID SONG_NAME	Hotel California Eagles			
SONG_ID SONG_NAME	7			
SONG_NAME				
SONG_NAME	SONG_ID			
	SONG_NAME			

SQL Plus	x + ~	-	o	×
SINGER_NAME				
MOVIE_ID				
8 Bont Stop Believin Journey 8				
SONG_ID				
SONG_NAME				
SINGER_NAME				
MOVIE_ID 9 Thriller Michael Jackson 9				
SONG_ID				
SONG_NAME				
SINGER_NAME				
MOVIE_ID 10 Wonderwall				

Wonderwall Oasis 10				
SONG_ID				
SONG_NAME		 		
SINGER_NAME				
MOVIE_ID				
11 Uptown Funk Mark Ronson 1	ft. Bruno Mars			
SONG_ID				
SONG_NAME		 		
SINGER_NAME		 		
MOVIE_ID				
Smooth Santana ft. 2	Rob Thomas			
SONG_ID				

🗟 SQL Plus X + V		×
SONG_ID		
SONG_NAME		
SINGER_NAME		
MOVIE_ID		
Livin on a Prayer Bon Jovi 3		
SONG_ID		
SONG_NAME		
SINGER_NAME		
MOVIE_ID		
14 Hello Adele H		
SONG_ID		
SONG_NAME		
SINGER_NAME		

🔄 SQL Plus 🛛 🗙	+ •	 o <u>×</u>
SONG_NAME		
SINGER_NAME		
MOVIE_ID		
I Will Always Love You Whitney Houston 5		
SONG_ID		
SONG_NAME		
SINGER NAME		
 MOVTE TD		
Every Breath You Take The Police 6		
SONG_ID		
SONG_NAME		
SINGER NAME		
MOV/TE TD		
HOVIE_ID		I
17		
SQL Plus X	+ •	 o _×_
Eye of the Tiger		
Survivor 7		
SONG_ID		
SONG_NAME		
SINGER_NAME		
MOVIE_ID		
18		
Nothing Else Matters Metallica 8		
SONG_ID		
SONG_NAME		
 SINGER NAME		
Let It Be		
The Beatles 9		
SONG_ID		
🖬 SQL Plus 🛛 🗙	+ •	 o ×
Metallica 8		
SONG_ID		
SONG_NAME		
SINGER_NAME		
MOVIE_ID		
19		
Let It Be The Beatles 9		
SONG_ID		
SONG_NAME		
SINGER NAME		
MOVIE ID		
20 What a Wonderful World Louis Armstrong 10		
20 rows selected.		

Genre:

```
CREATE TABLE Genre (
genre_id INT NOT NULL,
genre_name VARCHAR(255) NOT NULL,
PRIMARY KEY (genre_id)
);
```

SQL> DROP TABLE Genre CASCADE CONSTRAINTS;
Table dropped.
\$QL> ren +
SQL>rem Create Genre Table
SQL> SQL>CREATE TABLE Genre (
2 genre_id TNT NOT NULL, 3 genre_iname VARCHAR(255) NOT NULL,
4 PRIMARY KEY (genre_id) 5);
Table created.
sqL>
E sol Plus × + ▼
2 genre_id INT NOT NULL, 3 genre_name VARCHAR(255) NOT NULL, 4 PRIMARY KEY (genre_id) 5);
Table created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (1, 'Action');
1 row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (2, 'Comedy');
1 row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (3, 'Drama');
1 row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (4, 'Romance');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (S, 'Thriller');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (6, 'Adventure');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (7, 'Science Fiction');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (8, 'Fantasy');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (9, 'Musical');
l row created.
SQL> INSERT INTO Genre (genre_id, genre_name) VALUES (10, 'Animation');
l row created.
sor>

SQL Plus		×	$+$ \checkmark	-	o	×
SQL> select	* from	Genre;				
GENRE_ID						
GENRE_NAME						
1 Action						
2 Comedy						
3 Drama						
GENRE_ID						
GENRE_NAME						
4 Romance						
5 Thriller						
6 Adventure						
GENRE_ID						
GENRE_NAME						
7						

SQL Plus	× + ~			o ×
4 Romance				
5 Thriller				
6 Adventure				
GENRE_ID				
GENRE_NAME				
7 Science Fiction				
8 Fantasy				
9 Musical				
GENRE_ID				
GENRE_NAME		 		
10 Animation				
10 rows selected.				

Tagged With:

CREATE TABLE TaggedWith (movie_id INT NOT NULL, genre_id INT NOT NULL, PRIMARY KEY (movie_id, genre_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id), FOREIGN KEY (genre_id) REFERENCES Genre(genre_id)



В SQL Plus X + ✓ — Ø X
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (6, 5);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (11, 1);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (7, 5);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (8, 6);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (8, 7);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (13, 8);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (9, 6);
1 row created.
SQL> INSERT INTO TaggedWith (movie_id, genre_id) VALUES (9, 7);
1 row créated.
SQL> INSERT INTO TaggedMith (movie_id, genre_id) VALUES (14, 8);
1 row created.
SQL> INSERT INTO TaggedMith (movie_id, genre_id) VALUES (10, 1);
1 row created.
sQL> INSERT INTO TaggedHith (movie_id, genre_id) VALUES (10, 2);
1 row created.

SQL> select	* from Tag	ggedWith;	
MOVIE_ID	GENRE_ID		
1	4		
1	3		
20	4		
2	3		
12			
13			
6			
MOVIE_ID	GENRE_ID		
6	5		
11	1		
,	5		
8	7		
13	, 8		
9	6		
9	7		
14	8		
10	1		
10	2		
22 rows sel	ected.		

SiteLocation:

CREATE TABLE SiteLocation (location_id INT NOT NULL, location_name VARCHAR(255) NOT NULL, address VARCHAR(255) NOT NULL, PRIMARY KEY (location_id)

);

SQL> DROP TABLE SiteLocation CASCADE CONSTRAINTS;
Table dropped.
SQL> rem ↓
Table created. SQL>

🗊 SQL Plus 🛛 🗡		o ×
SQL> INSERT INTO SiteLocation (loca 2 VALUES (1, 'Central Park', '12	ration_id, location_name, address) 23 Main St. New York, WY 100359);	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (2, 'Lincoln Memorial',	ation_id, location_name, address) , '2 Lincoln Memorial Cir WW, Mashington, DC 20037');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (3, 'Golden Gate Bridge	ation_id, location_name, address) le', 'Golden Gate Bridge, San Francisco, CA 94129');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (4, 'The Bean', 'Cloud	ration_id, location_name, address) Gate, Chicago, IL 60601');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (5, 'Niagara Falls', 'N	ation_id, location_name, address) Niagara Falls State Park, Miagara Falls, NY 14303');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (6, 'Mount Rushmore', '	ation_id, location_name, address) '13000 SD-244, Keystone, SD 57751');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (7, 'The Alamo', '300 A	ation_id, location_name, address) Alamo Plaza, San Antonio, TX 78285');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (8, 'Grand Canyon', 'Gr	ation_id, location_name, address) irand Canyon Village, AZ 86023');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (9, 'Niagara Falls', 'N	ation_id, location_name, address) Niagara Falls, ON L2G 3Y9, Canada');	
1 row created.		
SQL> INSERT INTO SiteLocation (loca 2 VALUES (10, 'Redwood National	ation_id, location_name, address) .and State Parks', 'llll 2nd St, Crescent City, CA 95531');	
1 row created.		

SQL Plus	X + ~		o >	×
SQL> select * from B	uilding;			
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
1 MGM Studios studio				
2 Sony Pictures studio				
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
3 Pinewood Studios studio				
4 Warner Bros. Studios				
BUILDING_ID				
BUILDING_NAME				1
SQL Plus	x + ~	— ć) ×	
छ SQL Plus SQL> select * from S	x + v	- c) X	
छ SQL Plus SQL> select * from S: LOCATION_ID	x + ~	– c) X	
SQL Plus SQL> select * from S: LOCATION_ID 	x + v	— c) ×	
SQL Plus SQL> select * from S: LOCATION_TD LOCATION_NAME ADDRESS	x + v	– c	9 ×	
SQLPUS SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 Central Park 123 Main St, New York	x + v (teLocation; 	— c	× •	
SQLPUS SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 central Park 123 Main St, New Yord 2 Lincoln Memorial C: 2 Lincoln Memorial C:	× + ✓ (teLocation;	— c	× 13 ×	
SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 central Park 123 Main St, New Yorl 2 Lincoln Memorial 2 Lincoln Memorial C: LOCATION_ID	× + ✓ (telocation;	— c	x x	
SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 Central Park 123 Main St, New Yorl 2 Lincoln Memorial 2 Lincoln Memorial c: LOCATION_ID LOCATION_NAME	× + ✓ iteLocation;	— c) ×	
SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 Central Park 123 Main St, New Yorl 2 Lincoln Memorial C: LOCATION_ID LOCATION_NAME ADDRESS	x + v iteLocation; 	- c	• ×	
SQL> Select * from S: LOCATION_ID LOCATION_NAME ADDRESS Central Park 123 Main St, New Yorl 2 Lincoln Memorial 2 Lincoln Memorial LOCATION_NAME ADDRESS Golden Gate Bridge, S	<pre>x + v iteLocation; , NY 10019 ir NW, Washington, DC 20037 san Francisco, CA 94129</pre>	- c	. x	
SQL> Select * from S: LOCATION_ID LOCATION_NAME ADDRESS 123 Main St, New Yorl 2 Lincoln Memorial 2 Lincoln Memorial 2 Lincoln Memorial 2 LocATION_NAME ADDRESS Golden Gate Bridge, S Golden Gate Bridge, S The Bean	<pre>x + v iteLocation; </pre>	- c	• ×	
SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS 1 central Park 123 Main St, New York 2 Lincoln Memorial c: LOCATION_ID LOCATION_MAME ADDRESS 3 Golden Gate Bridge Golden Gate Bridge, s 4 The Bean LOCATION_ID	<pre>x + v iteLocation;</pre>	- c	9 ×	
SQL> select * from S: SQL> select * from S: LOCATION_ID LOCATION_NAME ADDRESS Central Park 123 Main St, New Yord Central Park 123 Main St, New Yord Lincoln Memorial C: LOCATION_ID LOCATION_NAME ADDRESS Golden Gate Bridge Golden Gate Bridge St Golden Gate Bridge St 4 The Bean LOCATION_ID LOCATION_NAME	<pre>x + v iteLocation;</pre>	- 6	X	

SQL Plus X + v	-	o	×
ADDRESS			
Cloud Gate, Chicago, IL 60601			
5 Niagara Falls Niagara Falls State Park, Niagara Falls, NY 14303			
Location_ID			
LOCATION_NAME			
ADDRESS			
Mount Rushmore 13000 SD-244, Keystone, SD 57751			
7 The Alamo 300 Alamo Plaza, San Antonio, TX 78205			
LOCATION_ID			
LOCATION_NAME			
ADDRESS			
8 Grand Canyon Grand Canyon Village, AZ 86023			I
SQL Plus X + V	- 6))	×
7 The Alamo 300 Alamo Plaza, San Antonio, TX 78205			
LOCATION_ID			
LOCATION_NAME			
ADDRESS			
S Grand Canyon Grand Canyon Village, AZ 86023			
9 Niagara Falls, ON L2G 3Y9, Canada			
LOCATION_ID			
LOCATION_NAME			
ADDRESS			
10 Redwood National and State Parks 1111 2nd St, Crescent City, CA 95531			
10 rows selected.			
sqL>			I

Building:

CREATE TABLE Building (building_id INT NOT NULL, building_name VARCHAR(255) NOT NULL, purpose VARCHAR(20) NOT NULL, location_id INT NOT NULL, PRIMARY KEY (building_id), FOREIGN KEY (location_id) REFERENCES SiteLocation(location_id), CONSTRAINT purpose_constraint_violated CHECK (purpose IN ('production', 'postproduction', 'office', 'studio', 'storage', 'visual effects', 'color grading', 'sound design', 'mixing')));

SQL> DROP TABLE Building CASCADE CONSTRAINTS;	
Table dropped.	
<pre>SQL> rem /</pre>	
Table created.	

🗊 SQL Plus

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (1, 'MGM Studios', 'studio', 1);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (2, 'Sony Pictures', 'studio', 2);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (3, 'Pinewood Studios', 'studio', 3);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (4, 'Warner Bros. Studios', 'studio', 4);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (5, 'Fox Studios', 'studio', 5);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (6, 'Universal Studios', 'studio', 6);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (7, 'Paramount Studios', 'studio', 7);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (8, 'ABC Studios', 'studio', 8);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (9, 'Columbia Pictures', 'studio', 9);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (10, 'Marvel Studios', 'studio', 10);

1 row created.

🖾 SQL Plus

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (11, 'DreamWorks Animation', 'visual effects', 1); 1 row created.

× + ~

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (12, 'Walt Disney Animation Studios', 'visual effects', 2);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (13, 'Industrial Light and Magic', 'visual effects', 3);

1 row created.

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (14, 'Rhythm and Hues Studios', 'visual effects', 4);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (15, 'Pixar Animation Studios', 'visual effects', 5);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (16, 'MPC Film', 'visual effects', 6);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (17, 'Double Negative', 'visual effects', 7);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (18, 'Sony Pictures Imageworks', 'visual effects', 8);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (19, 'Framestore', 'visual effects', 9);

SQL> INSERT INTO Building (building_id, building_name, purpose, location_id) 2 VALUES (20, 'Technicolor SA', 'color grading', 10);

SQL Plus	× + ~		9	
SQL> select * from Bu	uilding;			
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
1 MGM Studios studio				
2 Sony Pictures studio				
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
3 Pinewood Studios studio 4				
Warner Bros. Studios				
BUILDING_ID				
BUILDING_NAME				
SQL Plus	× + ~	- 9		×
BUILDING_NAME				
PURPOSE	LOCATION_ID			
 studio				
5 Fox Studios studio				
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
Universal Studios studio				
7 Paramount Studios studio				
BUILDING_ID				
BUILDING_NAME	LOCATION ID			
 8 APC Studios				1
studio	8			
SQL Plus			×	<
8 ABC Studios studio				
9 Columbia Pictures studio				
BUILDING_ID BUILDING_NAME				
PURPOSE	LOCATION_ID			
10 Marvel Studios studio	10			
11 DreamWorks Animation				
BUILDING_ID BUILDING_NAME				
PURPOSE	LOCATION_ID			
visual effects				
12 Walt Disney Animation visual effects	n Studios 2			

SQL Plus	× + ~	-	· 0	×
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
Industrial Light and visual effects	Magic 3			
14 Rhythm and Hues Stud: visual effects	ios 4			
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
15 Pixar Animation Stud: visual effects	 ios 5			
16 MPC Film visual effects	6			
BUILDING ID				
BUILDING NAME				
 	LOCATION ID			ı
SQL Plus	× + ~		Ø	×
16 MPC Film				
visual effects				
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
17 Double Negative visual effects				
18 Sony Pictures Imagewo	orks			
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
 visual effects	8			
19 Framestore visual effects	9			
20				
BUILDING_ID				
BUILDING_NAME				
SQL Plus				
17 Double Negative visual effects				
18 Sony Pictures Imagew	orks			
BUILDING_ID				
BUILDING_NAME				
PURPOSE	LOCATION_ID			
visual effects	8			
19 Framestore visual effects	9			
20				
BUILDING ID				
DURDOSE				
	LOCATION_ID			
Technicolor SA color grading	10			
20 rows selected.				
sqL>				

ShotAt:

CREATE TABLE ShotAt (location_id INT NOT NULL, movie_id INT NOT NULL, PRIMARY KEY (location_id, movie_id), FOREIGN KEY (location_id) REFERENCES SiteLocation(location_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id));

/ ,
SQL> DROP TABLE Shotat CASCADE CONSTRAINTS;
Table dropped.
SQL> rem +
Table created.
SQL>
🖾 SQL Plus X + V
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (1, 1);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (2, 1);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (2, 2);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (3, 2);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (4, 3);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (4, 4);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (5, 5);
1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (6, 6);

1 row created.
SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (7, 7);
1 row created.

SQL> INSERT INTO ShotAt (location_id, movie_id) VALUES (7, 8); 1 row created.

OUTPUT:

SOL> select	* from Shot
LOCATION_ID	MOVIE_ID
1	1
2	2
3	2
4	
4	
5	
6	
7	7
7	8
7	9
LOCATTON TO	MOVIE TO
8	10
9	11
10	12
10	13
10	14
10	15
10	16
10	17
10	18
10	19
21 rows sele	cted.
SQL>	

PostProductionDoneIn:

CREATE TABLE PostProductionDoneIn (movie_id INT NOT NULL, building_id INT NOT NULL, PRIMARY KEY (movie_id, building_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id), FOREIGN KEY (building_id) REFERENCES Building(building_id) SQL> DROP TABLE PostProductionDoneIn CASCADE CONSTRAINTS; Table dropped. SQL> rem SQL> rem | Create PostProductionDoneIn Table SQL> rem + SQL> SQL> CREATE TABLE PostProductionDoneIn (AIE TABLE POSTFICULETIONDONEIN (movie_id INT NOT NULL, building_id INT NOT NULL, PRIMARY KEY (movie_id, building_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id), FOREIGN KEY (building_id) REFERENCES Building(building_id) 2 3 4 Table created.



🖾 SQL Plus

SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (1, 1); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (2, 2); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (3, 3); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (4, 4); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (5, 5); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (6, 6); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (8, 8); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (9, 9); 1 row created. SQL> INSERT INTO PostProductionDoneIn (movie_id, building_id) VALUES (10, 10); 1 row created.

OUTPUT:





CREATE TABLE Employees (employee_id INT NOT NULL, employee_name VARCHAR(255) NOT NULL, designation VARCHAR(255) NOT NULL,

);

phone_number VARCHAR(10) NOT NULL UNIQUE,

PRIMARY KEY (employee_id),

CONSTRAINT designatoin_constraint_violated CHECK (

designation IN ('choreographers', 'security', 'sound engineer', 'makeup artist', 'electrician', 'janitor', 'manager')));

SQL> DROP TABLE Employees CASCADE CONSTRAINTS;
Table dropped.
<pre>SQL> rem +</pre>
Table created.
sõr>
SQL Plus × + ✓ − 0 ×
Table created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (101, 'John Doe', 'electrician', '1234567890');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (102, 'Jane Smith', 'makeup artist', '2345678901');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (103, 'Bob Johnson', 'janitor', '3456789012');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (104, 'Sara Lee', 'sound engineer', '4567890123');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (105, 'Mike Tyson', 'security', '5678901234');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (106, 'Lisa Brown', 'choreographer', '6789012345');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (107, 'David Kim', 'manager', '7890123456');
1 row created.
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (108, 'Jessica Lee', 'electrician', '8901234567');
1 row created.

S SOL Plus X + V	-	ø	×
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (189, 'Jason Lee', 'makeup artist', '9012345678');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (110, 'Emily Johnson', 'janitor', '0123456789');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (111, 'Alex Lee', 'sound engineer', '1234509876');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (112, 'Alice Kim', 'security', '2345609876');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (113, 'Brian Kim', 'choreographer', '3456709876');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (114, 'Cathy Lee', 'manager', '4567809876');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (115, 'Daniel Lee', 'electrician', '5678909876');			
1 row created.			
SQL> INSERT INTO Employees (employee_id, employee_name, designation, phone_number) 2 VALUES (116, 'Ellen Kim', 'makeup artist', '6789019876');			
1 row created.			

E SQL Plus × + -	o	
SQL> select * from Employees;		
EMPLOYEE_ID		
= PPLOYEE_NAME		
DESIGNATION		
PHONE_NUMB		
John Doe electrician		
1234567890		
EMPLOYEE_ID		
DESIGNATION		
PHONE NUMB		
Jane Smith makeun artist		
2345678901		
Enployte_NAme		
DESTONATION		
	a	\times
S SQL Plus X + V		×
© SQL Plus × + → DESIGNATION PHONE NUMB		×
S SQL Plus X + V DESIGNATION PHONE_NUMB 103		×
© sqLPlus × + ∨ DESIGNATION PHONE_NUMB 103 Bob Johnson		×
© SQL Plus × + ↓ DESIGNATION PHONE_NUHB 103 Bob Johnson janitor 3456789012		×
SQL Plus X + ~ DESIGNATION PHONE_NUMB 103 Bob Johnson janitor 3456789012		×
S SQL PRUS X + ~ DESIGNATION PHONE_NUMB 103 Bob Johnson janitor 3456789012 EMPLOYEE_ID EMPLOYEE_ID		×
S SQL PUIS X + DESIGNATION PHONE_NUMB 103 Bob Johnson janitor 34866789012 EMPLOYEE_ID EMPLOYEE_NAME		×
Sql Plus × + × DESIGNATION		×
Sql Plus × + × DESIGNATION PHONE_NUMB 103 Bob Johnson janitor 348679912		×
Sal Plus × + DESIGNATION		×
Sql Plus + - DESIGNATION PHONE_NUMB 103 Bob Johnson janitor 3456789012 PHOLYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NUMB 104 Sara Lee sound engineer \$50789123		×
Sql Plus + - PHONE_NUHB 103 Bob Johnson janitor 3456789012 EMPLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NUHB 104 Sara Leg sound engineer sound engineer		×
SqL Plus X + DESIGNATION		×
SqL Plus × PHOLESIGNATION PHOLESIGNATION PHOLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NUMB 104 Sara Lee Sourd engineer Sourd engineer Sourd engineer		×
SqL Plus X + DESIGNATION PHOLE_NURB 103 Bob Johnson janitor 3456789012 EMPLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NURB 104 Sara Lee Sound engineer Sound engineer Sound engineer Sound engineer Sound engineer DESIGNATION EMPLOYEE_ID EMPLOYEE_ID EMPLOYEE_ID EMPLOYEE_ID ENDIFIER DESIGNATION PHONE_NURB DESIGNATION EMPLOYEE_ID EMPLOYEE_ID ENDIFIER DESIGNATION EMPLOYEE_ID EM		×
SqL Plus X + X DESIGNATION PHONE_NURB 103 Bob Johnson janitor 3456789012 EMPLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NURB 104 Sara Lee Sound engineer Sound engineer Sound engineer Sound engineer Sound engineer Sound engineer DESIGNATION PHOVEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NURB DESIGNATION PHONE_NUR		×
SqL Plus X + DESIGNATION PHONE_NURB 103 Bob Johnson janitor 3456789012 EMPLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NURB 104 Sara Lee sound engineer 305 FMPLOYEE_ID EMPLOYEE_NAME DESIGNATION PHONE_NURB ENDIFYEE_NAME DESIGNATION PHONE_NURB DESIGNATION PHONE_NURB DESIGNATION PHONE_NURB 105 105 105		×

SQL Plus	X + •	- 1	א נ
security 5678901234			
EMPLOYEE TD			
PHONE_NUMB			
106 Lisa Brown choreographer 6789012345			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
107			
David Kim manager			
7890123456			
EMPLOYEE TO			
	x + ×	_	r ×
DESIGNATION			
PHONE_NUMB			
108			
electrician 8901234567			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
109			
Jason Lee makeup artist 9012345678			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
110			l
Emily Johnson			
SQL Plus	x + v		×
θ123456789			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
111			
sound engineer			
1254569670			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
112			
Alice Kim security			
2345609876			
EMPLOYEE_ID			
EMPLOYEE_NAME			l

SQL Plus ×	+ •		n x
DESIGNATION			
PHONE_NUMB			
113			
Brian Kim choreographer 3456709876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
114			
Cathy Lee manager 4567809876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
 115 Daniel Lee			l
		_	
SQL Plus ×			· ×
Daniel Lee electrician 5678909876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
116			
Ellen Kim makeup artist 6789019876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
117			
janitor			
7890129876			
EMPLOYEE_ID			I
🖾 SQL Plus 🛛 🗙		0	×
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
118 Grace Kim			
sound engineer 8901239876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
119 Henry Lee			
security 9012349876			
EMPLOYEE_ID			
EMPLOYEE_NAME			
DESIGNATION			
PHONE_NUMB			
120			

🗊 SQL Plus 🛛 🗙	+ •	-	o	×
sound engineer 8901239876				
EMPLOYEE_ID				
EMPLOYEE_NAME				
DESIGNATION				
PHONE_NUMB				
119 Henry Lee security 9012349876				
EMPLOYEE_ID				
EMPLOYEE_NAME				
DESIGNATION				
PHONE_NUMB				
120 Ivy Kim choreographer 0123456987				
20 rows selected.				
SQL>				

Payroll:

CREATE TABLE Payroll (payroll_id INT NOT NULL, salary FLOAT NOT NULL, employee_id INT NOT NULL, hours_worked FLOAT NOT NULL, PRIMARY KEY (payroll_id), FOREIGN KEY (employee_id) REFERENCES Employees(employee_id), CONSTRAINT salary_constraint_violated CHECK (salary > 0), CONSTRAINT hours_constraint_violated CHECK (hours_worked > 0 and hours_worked <= 40)):

//
SQL> DROP TABLE Payroll CASCADE CONSTRAINTS;
Table dropped.
SQL> rem +
SQL> SQL>REATE TABLE Payroll (2 payroll_id INT NOT NULL, 3 salary FLOAT NOT NULL,
4 employee_id_INT_NOT_NULL, 5 hours_morkef_LOAT_NOT_NULL, 6 PRIMARY_KEY_(payroll_id), 7 EFORFICN_KEY_(payroll_id)_DEFERENCES_Employee_id)
8 CONSTRAINT salary_constraint_violated CHECK (salary > 0) 9 CONSTRAINT hours_constraint_violated CHECK (hours_worked > 0 and hours_worked <= 40) 10);
Table created.
sqL>

OUTPUT:			
SQL> select *	from Pa	yroll;	
PAYROLL_ID	SALARY	EMPLOYEE_ID	HOURS_WORKED
1	5000	101	40
	6000	102	38.5
	4500	103	37
	5500	104	39.5
	4000	105	36
	6500	106	40
	4800	107	38.5
	5200	108	37
	5800	109	39.5
10	4200	110	36
11	5100	111	40
PAYROLL_ID	SALARY	EMPLOYEE_ID	HOURS_WORKED
12	5300	112	38.5
13	4700	113	37
14	5400	114	39.5
15	3900	115	36
16	5900	116	40
17	4600	117	38.5
18	5000	118	37
19	5700	119	39.5
20	4300	120	36

1 row	created.						
SQL> 2	INSERT INTO VALUES (10,	Payroll (payroll_id, s 4200.00, 110, 36.00);	alary, employee_id	, hours_worked)			
1 row	created.						
SQL> 2	INSERT INTO VALUES (11,	Payroll (payroll_id, s 5100.00, 111, 40.00);	alary, employee_id	, hours_worked)			
1 row	created.						
SQL> 2	INSERT INTO VALUES (12,	Payroll (payroll_id, s 5300.00, 112, 38.50);	alary, employee_id	, hours_worked)			
l row	created.						
SQL> 2	INSERT INTO VALUES (13,	Payroll (payroll_id, s 4700.00, 113, 37.00);	alary, employee_id	, hours_worked)			
1 row	created.						
SQL> 2	INSERT INTO VALUES (14,	Payroll (payroll_id, s 5400.00, 114, 39.50);	alary, employee_id	, hours_worked)			
1 row	1 row created.						
SQL> 2	INSERT INTO VALUES (15,	Payroll (payroll_id, s 3900.00, 115, 36.00);	alary, employee_id	, hours_worked)			

🖾 SQL Plus

1	row	created.	

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked)
2 VALUES (9, 5800.00, 109, 39.50);

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (16, 5900.00, 116, 40.00);

1 row created. SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (8, 5200.00, 108, 37.00);

1 row created. SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (7, 4800.00, 107, 38.50);

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (6, 6500.00, 106, 40.00);

1 row created.

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked)
2 VALUES (3, 4500.00, 103, 37.00);

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (1, 5000.00, 101, 40.00);

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (2, 6000.00, 102, 38.50);

1 row created.

SQL Plus

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (4, 5500.00, 104, 39.50);

1 row created.

SQL> INSERT INTO Payroll (payroll_id, salary, employee_id, hours_worked) 2 VALUES (5, 4000.00, 105, 36.00);

SponsoringCompany:

CREATE TABLE SponsoringCompany (company_id INT NOT NULL, company_name VARCHAR(255) NOT NULL UNIQUE, PRIMARY KEY (company_id)

):
SQL> DROP TABLE SponsoringCompany CASCADE CONSTRAINTS;
Table dropped.
SQL> rem ++ SQL> rem Create SponsoringCompany Table SQL> rem +
Table created.
SQL>
SQL Plus × + →
Table created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (1, 'Universal Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (2, 'Warner Bros. Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany_id, company_name) 2 VALUES (3, 'Walt Disney Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (4, 'Paramount Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (5, '20th Century Fox');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (6, 'Sony Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (7, 'Lionsgate');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (8, 'DreamWorks Pictures');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (9, 'Miramax');
1 row created.
SQL> INSERT INTO SponsoringCompany (company_id, company_name) 2 VALUES (10, 'New Line Cinema');

1 row created.

SqL Plus × + -	
SQL> select * from SponsoringCompany ;	
COMPANY_ID	
COMPANY_NAME	
l Universal Pictures	
2 Warner Bros. Pictures	
3 Walt Disney Pictures	
COMPANY_ID	
COMPANY_NAME	
4 Paramount Pictures	
5 20th Century Fox	
6 Sony Pictures	
COMPANY_ID	
COMPANY_NAME	
7	
S SQL Plus X + V	- o ×
© sqLPlus × + ↓ 4 Paramount Pictures	- o x
© sqLPlus × + ∽ 4 Paramount Pictures 20th Century Fox	- o x
E squ Plus × + ~ 4 Paramount Pictures 5 20th Century Fox 6 Sony Pictures	- a x
E sourlus × + × Paramount Pictures 20th Century Fox 6 Sony Pictures COMPANY_ID	- o ×
E SQLPHus × + × Paramount Pictures 20th Century Fox 6 Sony Pictures COMPANY_ID COMPANY_NAME	- o ×
Sourius × + × A Paramount Pictures 5 20th Century Fox 6 Sony Pictures COMPANY_ID COMPANY_ID T COMPANY_NAME 7 Lionsgate	- a x
E sourius × + × Paramount Pictures 5 20th Century Fox 6 Sony Pictures COMPANY_ID COMPANY_NAME 7 Lionsgate 8 DreamWorks Pictures	- o ×
Sourius × + × A Paramount Pictures 20th Centry Fox Sony Pictures COMPANY_ID COMPANY_NAME 7 Lionsgate Pranworks Pictures 9 Miramax	– a ×
Souries × + × Paramount Pictures Soft Century Fox Sony Pictures COMPANY_ID COMPANY_INAME PreamWorks Pictures P P COMPANY_ID COM	– o ×
Sourius × + × Paramount Pictures Sourius So	– o ×
Sourius × + × Paramount Pictures Soft Centry Fox COMPANY_ID COMPANY_NAME T T Company_NAME P Miramax COMPANY_ID Company_NAME P Miramax COMPANY_ID Company_NAME P Miramax COMPANY_ID Company_NAME Company_NAME	- o x
sourius × + × Paramount Pictures Sony Pictures COMPANY_ID COMPANY_NAME P r P r P r f	- o ×

Manages:

CREATE TABLE Manages (employee_id INT NOT NULL, location_id INT NOT NULL, PRIMARY KEY (employee_id, location_id), FOREIGN KEY (employee_id) REFERENCES Employees(employee_id), FOREIGN KEY (location_id) REFERENCES SiteLocation(location_id)

;
SQL> DROP TABLE Manages CASCADE CONSTRAINTS;
Table dropped.
SQL> rem +
SQL> CHENT TABLE Manages (2 Ameril avea id THT NOT NULL
z tapicjic_izi zm no noch, 3 location_id INT NOT NULL, 4 PRIMARY KEY (employee_id, location_id), 5 FORTEW KEY (employee_id, DEFERENCES Frantauros id)
6 FOREIGN KEY (location_id) REFERENCES SiteLocation(location_id) 7);
Table created.
sqL>

😨 SQL Phus X +		×
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (101, 1);		
l row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (102, 2);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (103, 3);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (104, 4);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (105, 5);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (106, 6);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (107, 7);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (108, 8);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (109, 9);		
1 row created.		
SQL> INSERT INTO Manages (employee_id, location_id) VALUES (110, 10);		
1 row created.		



SQL> select	* from Mana
EMPLOYEE TO	
101	1
102	2
103	3
104	L
105	5
106	e
107	5
108	8
109	ç
110	16
101	2
EMPLOYEE_ID	LOCATION_ID
100	
102	
103	
104	
105	
100	5
108	c
109	16
110	1
20 rows sel	ected.
SQL>	

Artist:

CREATE TABLE Artist (artist_id INT PRIMARY KEY, artist_name VARCHAR(255) NOT NULL, date_of_birth DATE NOT NULL, gender VARCHAR(1) NOT NULL, CONSTRAINT gender_constraint_violated CHECK (gender in ('M', 'F', 'T'))

);	
SQL> DROP TABLE Artist CASCADE CONSTRAINTS;	
Table dropped.	
SQL> rem Create Artist Table SQL> rem Create Artist Table SQL> rem +	
Table created.	
squ>	

🖾 SQL Plus X + 🗸		- 0	×
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (1, 'Tom Hanks', to_date('07-09-1956', 'MM-DD-VYYY'), 'M');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (2, 'Meryl Streep', to_date('06-22-1949', 'MM-DD-YYYY'), 'F');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (3, 'Denzel Washington', to_date('12-28-1954', 'MM-DD-YYYY'), 'M');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (4, 'Scarlett Johansson', to_date('11-22-1984', 'MM-OD-YYYY'), 'F');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (5, 'Robert De Niro', to_date('08-17-1943', 'WM-DD-YYYY'), 'M');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (6, 'Natalie Portman', to_date('06-09-1981', 'NM-DD-YYYY'), 'F');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (7, 'Leonardo DiCaprio', to_date('11-11-1974', 'MM-DD-VYYY'), 'M');			
1 row created.			
SQL> INSERT INTO Artist (artist_id, artist_name, date_of_birth, gender) VALUES 2 (8, 'Saoirse Ronan', to_date('04-12-1994', 'MM-DD-YYYY'), 'F');			
1 row created.			

SQL Plus	× + -	-	٥	×
SQL> select * from Art:	ist;			
ARTIST_ID				
ARTIST_NAME				
DATE_OF_B G				
2 Meryl Streep 22-JUN-49 F				
ARTIST_ID				
ARTIST_NAME				
DATE_OF_B G				
3 Denzel Washington 28-DEC-54 M				
4 Scarlett Johansson				
ARTIST_ID				
ARTIST_NAME				1
DATE_OF_B G				

			• · ·
EN SQL Plus			~ ~
22-NOV-84 F			
5 Robert De Niro			
17-AUG-43 M			
6			
ARTIST_ID			
ARTIST_NAME			
DATE_OF_B G			
Natalie Portman			
09-JUN-81 F			
7			
Leonardo DiCaprio 11-NOV-74 M			
ARTIST_ID			
ARTIST_NAME			
DATE_OF_B G			
8			
Saoirse Ronan 12-APR-94 F			
9			
	× + v		
Brad Ditt		c	~ ~
18-DEC-63 M			
ARTIST_ID			
ARTIST_NAME			
DATE OF B G			
10			
Emma Stone 06-NOV-88 F			
11			
George Clooney			
ARTIST_ID			
ARTIST_NAME			
DATE_OF_B G			
 06-MAY-61 M			
12			
Anne Hathaway			
12-NOV-82 F			
13			
ARTIST_ID			
ARTIST_NAME			1
	V + 10		
SQL PIUS			
12			
12-NOV-82 F			
13			
ARTIST_ID			
DATE_OF_B G			
Johnny Depp 09-JUN-63 M			
14			
Keira Knightley			
20 MAK-05 F			
ARTIST_ID			
ARTIST_NAME			
15 Dwayne Johnson			
02-MAY-72 M			
15 rows selected.			
col s			

GetsPaidBy:

CREATE TABLE getsPaidBy (artist_id INT NOT NULL, company_id INT NOT NULL, PRIMARY KEY (artist_id, company_id), FOREIGN KEY (artist_id) REFERENCES Artist(artist_id), FOREIGN KEY (company_id) REFERENCES SponsoringCompany(company_id)); SQL> DROP TABLE getsPaidBy CASCADE CONSTRAINTS; Table dropped. SQL> rem + SQL> rem | Create getsPaidBy Table SQL> rem + SQL> SQL> CREATE TABLE getsPaidBy (2 artist_id THT NOT NULL, 3 company_id THT NOT NULL, 4 PRIMARY KEY (artist_id), 5 FOREIGN KEY (artist_id) REFERENCES Artist(artist_id), 6 FOREIGN KEY (company_id) REFERENCES SponsoringCompany(company_id) 7); Table created. SQL> | 🖾 SQL Plus × + ~ SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (1, 4); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (1, 5); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (2, 3); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (2, 4); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (2, 5); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (3, 2); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (3, 3); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (4, 1); 1 row created.

SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (4, 2); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (4, 3); 1 row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (4, 4);

l row created. SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (5, 1);

1 row created.

SQL> INSERT INTO getsPaidBy (artist_id, company_id) VALUES (5, 2); 1 row created.

OUTPUT:

501 > coloct	t from as	at a bail day.
syr> serect	- * +rom ge	ecseatudy,
ARTIST_ID	COMPANY_ID	
	4	4
	5	Ś.
2	3	
2	4	4
2	5	6
3	1	
3	2	
3	3	
3	4	
4	1	
	-	
ARTIST_ID	COMPANY_ID	
	3	
	4	4
	1	
5	2	
15 nows sol	ostod	
15 rows set	lected.	
501 >		
245		

Produces:

CREATE TABLE Produces (company_id INT NOT NULL, movie_id INT NOT NULL, PRIMARY KEY (company_id, movie_id), FOREIGN KEY (company_id) REFERENCES SponsoringCompany(company_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id)

);					
SQL> DROP TABLE Produces CASCADE CONSTRAINTS;					
Table dropped.					
<pre>SQL> rem ++ SQL> rem Create Produces Table SQL> rem ++ SQL> SQL> CREATE TABLE Produces (2 company_id INT NOT NULL, 3 movie_id INT NOT NULL, 4 PRIMARY KEY (company_id, movie_id), 5 FOREIGN KEY (company_id) REFERENCES SponsoringCompany(company_id), 6 FOREIGN KEY (movie_id) REFERENCES Movie(movie_id) 7);</pre>					
Table created.					
sqL>					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (1, 1);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (2, 3);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (3, 6);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (4, 9);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (5, 12);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (6, 14);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (7, 16);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (8, 18);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (9, 19);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (10, 20);					
1 row created.					
SQL> INSERT INTO Produces (company_id, movie_id) VALUES (2, 1);					

🖾 SQL Plus X +			×
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (2, 1);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (3, 2);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (4, 3);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (5, 4);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (6, 5);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (7, 6);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (8, 7);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (9, 8);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (10, 9);		
1 row created.			
SQL> INSERT INTO Produces (comp	npany_id, movie_id) VALUES (1, 10);		
1 row created.			
SQL>			

SQL> select	* from Prod	uces;
COMPANY_ID	MOVIE_ID	
1	1	
2	3	
3	6	
	12	
	14	
7	16	
8	18	
	19	
10	20	
2		
COMPANY_ID	MOVIE_ID	
2		
	2	
4		
6	5	
7	6	
8	7	
9	8	
10		
	10	
20 rows sele	ected.	
SOL>		

ActsIn:

CREATE TABLE ActsIn (movie_id INT NOT NULL, artist_id INT NOT NULL, PRIMARY KEY (movie_id, artist_id), FOREIGN KEY (movie_id) REFERENCES Movie(movie_id), FOREIGN KEY (artist_id) REFERENCES Artist(artist_id));

Table dropped.
SQL>rem [Create ActsIn Table SQL>rem
SQL>

⊡ SQLPlus × + v	0	×
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (1, 1);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (2, 2);		
i row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (3, 3);		
1 row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (4, 4);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (5, 5);		
1 row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (6, 6);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (7, 7);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (8, 8);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (9, 9);		
l row created.		
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (10, 10);		
l row created.		

SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (12, 15);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (7, 14);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (19, 13);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (2, 12);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (15, 13);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (4, 10);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (8, 11);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (18, 12);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (13, 12);	
1 row created.	
SQL> INSERT INTO ActsIn (movie_id, artist_id) VALUES (5, 12);	
1 row created.	
sqL>	

SQL> select	* from Ac	itsin;
MOVIE_ID	ARTIST_ID	
1	1	
2	2	
3	3	
4		
5	5	
6		
7		
8		
9		
10	10	
12	15	
MOVIE_ID	ARTIST_ID	
7	14	
19	13	
2	12	
15	13	
4	10	
8	11	
18	12	
13	12	
5	12	
20 rows sel	.ected.	
501 2		
2962		

Individual Contribution:

Here I have created the tables Buildings and Movie-Script-Inventory, Included the foreign key(location_id), primary key(building_id), check(purpose), and NotNull constraints. I have inserted 20 tuples into the buildings tables and also 10 tuples into movie_script_inventory table.

Obtained the tables according to inserted values in both tables.