Term Final Report

Group 4 Members
Rawan Alsabeh
Shenrui Guan
Xiaoli Lin
Nghi Nguyen
Jane Yun

PART 1. Description of Netflix's Business Rules

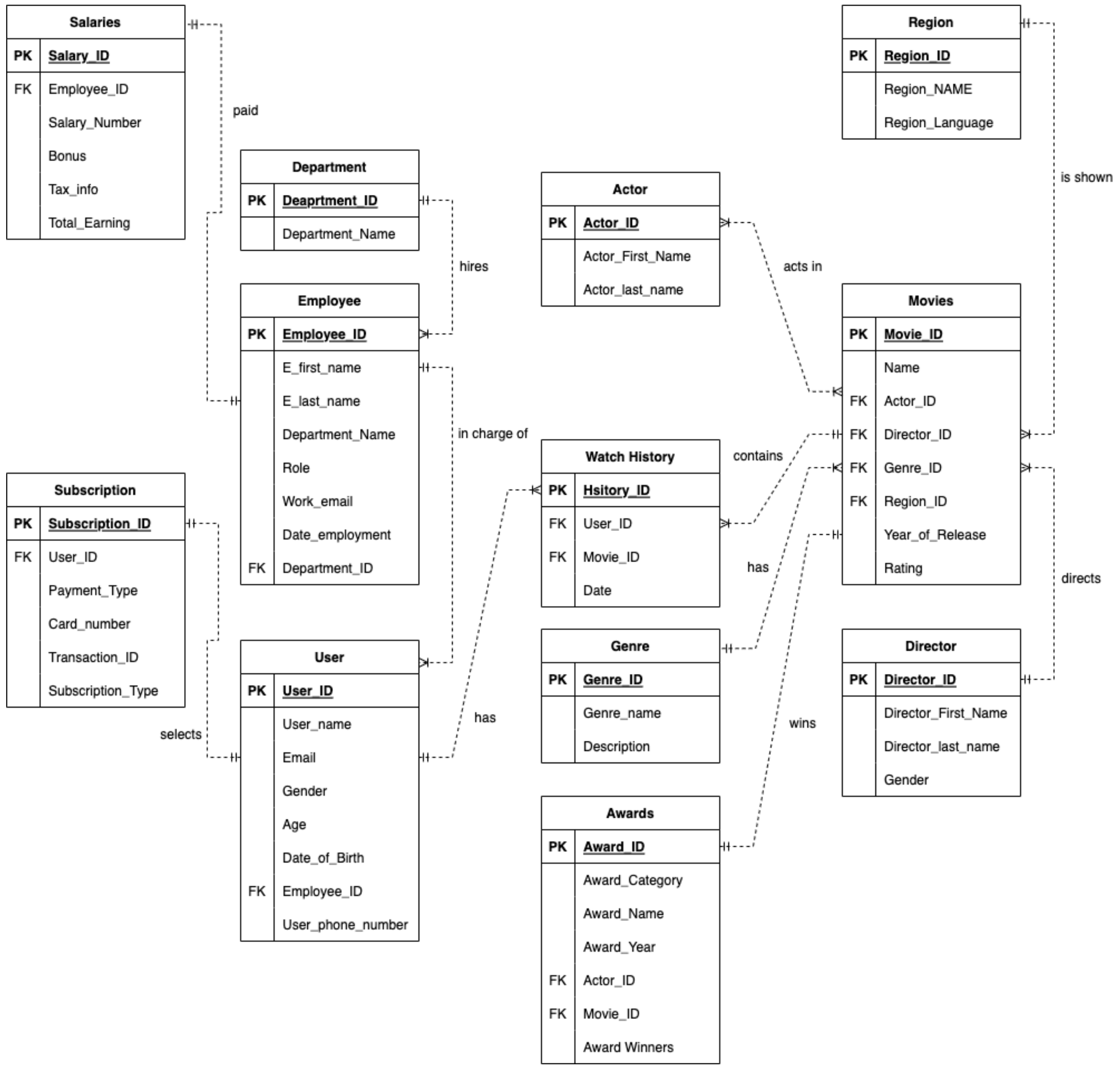
The ERD we created below is based on the management system of the users from Netflix. Netflix streams movies with multiple attributes to categorize (actors, region, genre, awards, and directors) and provides service to the users through subscriptions. Netflix also records users' watching history to analyze their watching patterns. We assume that their employees are assigned to manage the data of specific users. Each employee has a corresponding salary record managed by Netflix.

1. The Movies Table includes all the information about the movies. A movie can have multiple actors, one region, many watching histories, one genre, many awards, and one director. Movie_ID is the PK. The table includes names of the movie, actor_ID (FK to Actors table), director_ID (FK to Directors table), genre_ID (FK to Genre table), region_id (FK to Region table), year_of_release, and rating.
2. The Actor Table includes actor_ID as the PK, the first_name, and the last_name of the actor. Many actors could star in one or more movies.
3. The Region Table has region_ID as the PK, including region_name and region_language. A movie could have only one region, while a region can be tagged to many movies.
4. The Genre Table's PK is Genre_ID. There are names and descriptions of the genre on the Table. We assume one genre can be assigned to many movies, but one movie only has one genre.
5. The Awards Table uses award_ID as the PK. Movie_ID and actor_ID serve as the FK. Besides, the table comprises award_category, award_name, and award_year. One movie might have multiple awards, but each award is only issued to one movie.
6. The Directors Table's PK is director_id. It also records the director's first_name, last_name as well as gender. We assume one movie only has one director, while a director could direct many movies.
7. The Watch_History Table has history_ID as a PK. A user can have many watching histories, but one watching_history is linked to a unique user. Each history_ID is also linked to only one movie_ID. The date_time of watching is also recorded in the Table.
8. The User Table uses user_ID as the PK. The FK is employee_ID. The table includes the personal information about the users. A user could have multiple watching histories. A watching history could only belong to one user. Each user has one unique subscription. In addition, the user's data is managed by one employee, while an employee could manage many users.
9. The Subscription Table has subscription_ID as the PK and the user_ID as the FK. The table includes all the subscription information about the users, including Payment_Type,

Card_Number, Transaction_ID and Subscription_Types. One subscription_ID belongs to only one user.

10. The Employee Table includes all the information about the employees. The table uses employee_ID as the PK and Department_ID as the FK. One employee should take charge of the data of multiple users, but each user is assigned to only one employee. Moreover, one employee belongs to only one department
11. The Salaries Table has salary_ID as the PK and employee_ID as the FK. One salary_ID and employee_ID are in a one-to-one relationship.
12. The Department Tables has Department_ID as the PK, including Department_name.

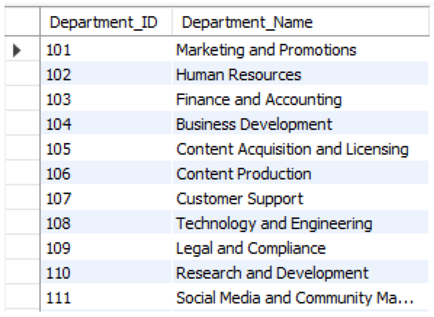
PART 2. ERD



PART 3. SQL Statements to Create Tables and Populate Tables to Load Data

1. **Department Table**

```
CREATE TABLE Department (
   Department_ID INT NOT NULL PRIMARY KEY,
   Department_Name VARCHAR(300) NOT NULL
);

INSERT INTO Department (Department_ID, Department_Name)
VALUES
(101, 'Marketing and Promotions'),
(102, 'Human Resources'),
(103, 'Finance and Accounting'),
(104, 'Business Development'),
(105, 'Content Acquisition and Licensing'),
(106, 'Content Production'),
(107, 'Customer Support'),
(108, 'Technology and Engineering'),
(109, 'Legal and Compliance'),
(110, 'Research and Development'),
(111, 'Social Media and Community Management'),
(112, 'Information Security'),
(113, 'Data Analytics'),
(114, 'Public Relations'),
(115, 'Risk Management'),
(116, 'Partnerships'),
(117, 'Data Privacy and Security'),
(118, 'IT Support'),
(119, 'Diversity and Inclusion'),
(120, 'Merger and Acquisitions');
```



Department_ID	Department_Name
101	Marketing and Promotions
102	Human Resources
103	Finance and Accounting
104	Business Development
105	Content Acquisition and Licensing
106	Content Production
107	Customer Support
108	Technology and Engineering
109	Legal and Compliance
110	Research and Development
111	Social Media and Community Ma...

2. Employee Table

```
Create Table Employee (
Employee_ID Int not null,
E_first_name varchar(22) not null,
E_last_name varchar(22) not null,
Department_Name varchar(300) not null,
Role varchar(50) not null,
Work_Email varchar(50) not null,
Date_Employment date not null,
Department_ID int not null ,
primary key (Employee_ID),
key Department_ID (Department_ID),
foreign key (Department_ID) references Department (Department_ID)
);


INSERT INTO employee (Employee_ID, E_first_name, E_last_name,
Department_name, Role, Work_Email, Date_Employment, Department_ID)
VALUES
(3427, 'Uriah', 'Bridges', 'Marketing and Promotions', 'Marketing Specialist',
'uriah.bridges@netflix.com', '2010-06-17', 101),
(3428, 'Paula', 'Small', 'Human Resources', 'Advisor', 'paula.small@netflix.com', '2007-
06-18', 102),
(3429, 'Edward', 'Buck', 'Finance and Accounting', 'Financial Analyst',
'edward.buck@netflix.com', '2012-06-20', 103),
```

(3430, 'Michael', 'Riordan', 'Business Development', 'Business Analyst', 'michael.riordan@netflix.com', '2012-06-20', 104),

(3431, 'Jasmine', 'Onque', 'Content Acquisition and Licensing', 'Manager', 'jasmine.onque@netflix.com', '2013-07-20', 105),

(3432, 'Maruk', 'Fraval', 'Content Production', 'Producer', 'maruk.fraval@netflix.com', '2011-06-21', 106),

(3433, 'Latia', 'Costa', 'Customer Support', 'Customer Service', 'latia.costa@netflix.com', '2013-06-21', 107),

(3434, 'Sharlene', 'Terry', 'Technology and Engineering', 'IT Support', 'sharlene.terry@netflix.com', '2015-06-21', 108),

(3435, 'Jac', 'McKinzie', 'Legal and Compliance', 'Lawyer', 'jac.mckinzie@netflix.com', '2015-07-21', 109),

(3436, 'Joseph', 'Martins', 'Research and Development', 'Developer', 'joseph.martins@netflix.com', '2015-09-01', 110),

(3437, 'Myriam', 'Givens', 'Social Media and Community Management', 'Specialist', 'myriam.givens@netflix.com', '2013-09-01', 111),

(3438, 'Dheepa', 'Nguyen', 'Information Security', 'Manager', 'dheepa.nguyen@netflix.com', '2013-07-20', 112),

(3439, 'Bartholemew', 'Khemmich', 'Data Analytics', 'Analyst', 'khemmich@netflix.com', '2012-07-21', 113),

(3440, 'Xana', 'Potts', 'Public Relations', 'Specialist', 'xana.potts@netflix.com', '2017-07-25', 114),

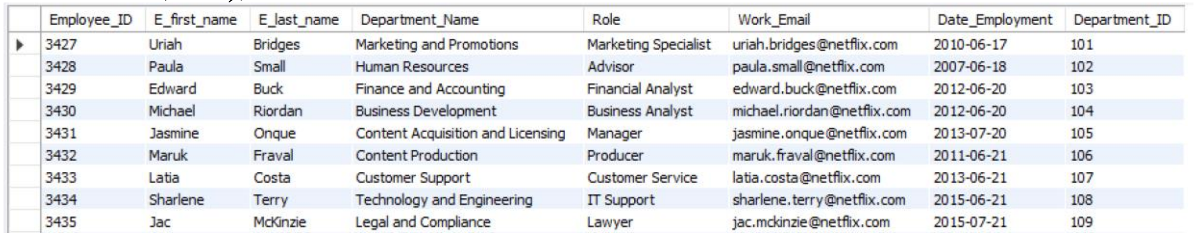
(3441, 'Prater','Jeremy', 'Risk Management', 'Analyst', 'prater.jeremy@netflix.com', '2017-07-04', 115),

(3442, 'Kaylah', 'Moon', 'Partnerships', 'Advisor', 'kaylah.moon@netflix.com', '2016-09-15', 116),

(3443, 'Kristen', 'Tate', 'Data Privacy and Security', 'Cybersecurity', 'kristen.tate@netflix.com', '2017-07-20', 117),

(3444, 'Bobby', 'Rodgers', 'Content Strategy', 'Strategist', 'bobby.rodgers@netflix.com', '2011-06-20', 118),

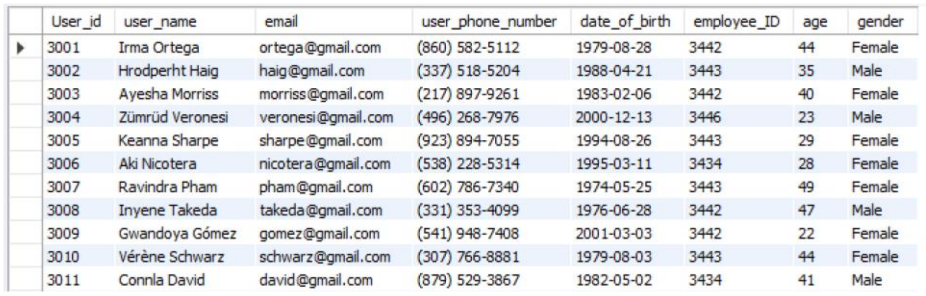
(3445, 'Reid', 'Park', 'Diversity and Inclusion', 'Advisor', 'reid.park@netflix.com', '2017-07-20', 119), (3446, 'John', 'Divad', 'Merger and Acquisitions', 'Manager', 'John.Dived@netflix.com', '2014-06-20', 120);



Employee_ID	E_first_name	E_last_name	Department_Name	Role	Work_Email	Date_Employment	Department_ID
3427	Uriah	Bridges	Marketing and Promotions	Marketing Specialist	uriah.bridges@netflix.com	2010-06-17	101
3428	Paula	Small	Human Resources	Advisor	paula.small@netflix.com	2007-06-18	102
3429	Edward	Buck	Finance and Accounting	Financial Analyst	edward.buck@netflix.com	2012-06-20	103
3430	Michael	Riordan	Business Development	Business Analyst	michael.riordan@netflix.com	2012-06-20	104
3431	Jasmine	Onque	Content Acquisition and Licensing	Manager	jasmine.onque@netflix.com	2013-07-20	105
3432	Maruk	Fraval	Content Production	Producer	maruk.fraval@netflix.com	2011-06-21	106
3433	Latia	Costa	Customer Support	Customer Service	latia.costa@netflix.com	2013-06-21	107
3434	Sharlene	Terry	Technology and Engineering	IT Support	sharlene.terry@netflix.com	2015-06-21	108
3435	Jac	McKinzie	Legal and Compliance	Lawyer	jac.mckinzie@netflix.com	2015-07-21	109

3. Users Table

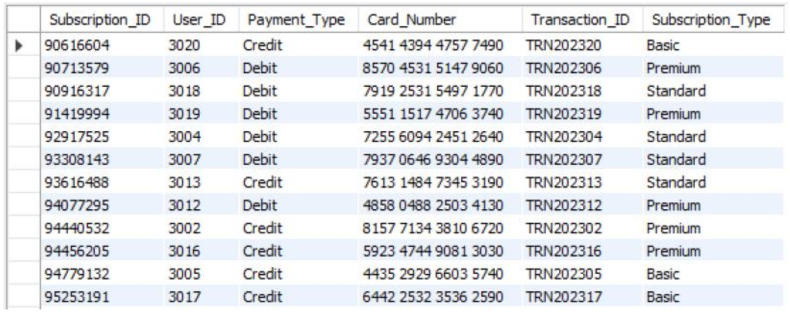
```
CREATE TABLE Users(

User_id int(11) NOT NULL,

user_name varchar(100) NOT NULL,

email varchar(100) NOT NULL,

user_phone_number varchar(50) NOT NULL,

date_of_birth date NOT NULL,

employee_ID int(11) NOT NULL,

age smallint NOT NULL,

gender varchar(20) NOT NULL,

PRIMARY KEY (user_id),

KEY employee_ID (employee_ID),

FOREIGN KEY (employee_ID) REFERENCES employee (employee_ID)

);


INSERT INTO Users (User_ID, User_name, Email, User_Phone_Number, Date_of_Birth, Employee_ID, Age, Gender)

VALUES

(3001, 'Irma Ortega', 'ortega@gmail.com', '(860) 582-5112', '1979-08-28', 3442, 44, 'Female'),

(3002, 'Hrodperht Haig', 'haig@gmail.com', '(337) 518-5204', '1988-04-21', 3443, 35, 'Male'),

(3003, 'Ayesha Morriss', 'morriss@gmail.com', '(217) 897-9261', '1983-02-06', 3442, 40, 'Female'),

(3004, 'Zümrüd Veronesi', 'veronesi@gmail.com', '(496) 268-7976', '2000-12-13', 3446, 23, 'Male'),
(3005, 'Keanna Sharpe', 'sharpe@gmail.com', '(923) 894-7055', '1994-08-26', 3443, 29, 'Female'),
(3006, 'Aki Nicotera', 'nicotera@gmail.com', '(538) 228-5314', '1995-03-11', 3434, 28, 'Female'),
(3007, 'Ravindra Pham', 'pham@gmail.com', '(602) 786-7340', '1974-05-25', 3443, 49, 'Female'),
(3008, 'Inyene Takeda', 'takeda@gmail.com', '(331) 353-4099', '1976-06-28', 3442, 47, 'Male'),
(3009, 'Gwandoya Gómez', 'gomez@gmail.com', '(541) 948-7408', '2001-03-03', 3442, 22, 'Female'),
(3010, 'Vérène Schwarz', 'schwarz@gmail.com', '(307) 766-8881', '1979-08-03', 3443, 44, 'Female'),
(3011, 'Connla David', 'david@gmail.com', '(879) 529-3867', '1982-05-02', 3434, 41, 'Male'),
(3012, 'Manoj Harrison', 'harrison@gmail.com', '(311) 825-3319', '1982-08-05', 3446, 41, 'Male'),
(3013, 'Lucas Gheorghe', 'gheorghe@gmail.com', '(889) 317-8701', '1990-09-01', 3443, 33, 'Male'),
(3014, 'Reidar Bulle', 'bulle@gmail.com', '(210) 782-7025', '1984-09-22', 3442, 39, 'Female'),
(3015, 'Sidónio Ruane', 'ruane@gmail.com', '(808) 229-9236', '1998-09-26', 3434, 25, 'Female'),
(3016, 'Aimé Melnyk', 'melnyk@gmail.com', '(673) 830-1055', '1979-10-26', 3442, 44, 'Female'),
(3017, 'Reshma Poppins', 'poppins@gmail.com', '(870) 665-3543', '1989-12-31', 3442, 34, 'Female'),
(3018, 'Uma Rocchi', 'rocchi@gmail.com', '(891) 282-8200', '1984-05-13', 3446, 39, 'Male'),
(3019, 'Anton Mitchell', 'mitchell@gmail.com', '(946) 489-5071', '1981-06-25', 3442, 42, 'Male'),
(3020, 'Padrig Vacík', 'vacik@gmail.com', '(886) 286-8458', '1975-10-10', 3434, 48, 'Male');
```



User_id	user_name	email	user_phone_number	date_of_birth	employee_ID	age	gender
3001	Irma Ortega	ortega@gmail.com	(860) 582-5112	1979-08-28	3442	44	Female
3002	Hrodperht Haig	haig@gmail.com	(337) 518-5204	1988-04-21	3443	35	Male
3003	Ayesha Morriss	morriss@gmail.com	(217) 897-9261	1983-02-06	3442	40	Female
3004	Zümrüd Veronesi	veronesi@gmail.com	(496) 268-7976	2000-12-13	3446	23	Male
3005	Keanna Sharpe	sharpe@gmail.com	(923) 894-7055	1994-08-26	3443	29	Female
3006	Aki Nicotera	nicotera@gmail.com	(538) 228-5314	1995-03-11	3434	28	Female
3007	Ravindra Pham	pham@gmail.com	(602) 786-7340	1974-05-25	3443	49	Female
3008	Inyene Takeda	takeda@gmail.com	(331) 353-4099	1976-06-28	3442	47	Male
3009	Gwandoya Gómez	gomez@gmail.com	(541) 948-7408	2001-03-03	3442	22	Female
3010	Vérène Schwarz	schwarz@gmail.com	(307) 766-8881	1979-08-03	3443	44	Female
3011	Connla David	david@gmail.com	(879) 529-3867	1982-05-02	3434	41	Male

4. Subscription Table

```
CREATE TABLE Subscription (
    Subscription_ID INT NOT NULL,
    User_ID INT NOT NULL,
    Payment_Type VARCHAR(20) NOT NULL,
    Card_Number CHAR(20) NOT NULL,
    Transaction_ID VARCHAR(20) NOT NULL,
    Subscription_Type VARCHAR(20) NOT NULL,
    PRIMARY KEY (Subscription_ID),
    KEY User_ID (User_ID),
    FOREIGN KEY (User_ID) REFERENCES Users (User_ID)
);


INSERT INTO Subscription (Subscription_ID, User_ID, Payment_Type, Card_Number,
Transaction_ID, Subscription_Type)
VALUES
(98811051, 3001, 'Debit', '5255 0186 4399 5570', 'TRN202301', 'Basic'),
(94440532, 3002, 'Credit', '8157 7134 3810 6720', 'TRN202302', 'Premium'),
(95816337, 3003, 'Credit', '5818 3366 1396 8370', 'TRN202303', 'Standard'),
(92917525, 3004, 'Debit', '7255 6094 2451 2640', 'TRN202304', 'Standard'),
(94779132, 3005, 'Credit', '4435 2929 6603 5740', 'TRN202305', 'Basic'),
(90713579, 3006, 'Debit', '8570 4531 5147 9060', 'TRN202306', 'Premium'),
(93308143, 3007, 'Debit', '7937 0646 9304 4890', 'TRN202307', 'Standard'),
(97186702, 3008, 'Debit', '4467 0562 8005 7740', 'TRN202308', 'Basic'),
(97262726, 3009, 'Credit', '8660 5743 6554 2650', 'TRN202309', 'Standard'),
(96160412, 3010, 'Credit', '5755 2376 7997 7860', 'TRN202310', 'Premium'),
(96717778, 3011, 'Debit', '8982 4868 7242 4110', 'TRN202311', 'Basic'),
(94077295, 3012, 'Debit', '4858 0488 2503 4130', 'TRN202312', 'Premium'),
(93616488, 3013, 'Credit', '7613 1484 7345 3190', 'TRN202313', 'Standard'),
(98455308, 3014, 'Credit', '7286 4072 6722 5480', 'TRN202314', 'Basic'),
(95918743, 3015, 'Credit', '7030 3916 8004 6220', 'TRN202315', 'Standard'),
(94456205, 3016, 'Credit', '5923 4744 9081 3030', 'TRN202316', 'Premium'),
(95253191, 3017, 'Credit', '6442 2532 3536 2590', 'TRN202317', 'Basic'),
(90916317, 3018, 'Debit', '7919 2531 5497 1770', 'TRN202318', 'Standard'),
(91419994, 3019, 'Debit', '5551 1517 4706 3740', 'TRN202319', 'Premium'),
(90616604, 3020, 'Credit', '4541 4394 4757 7490', 'TRN202320', 'Basic');
```

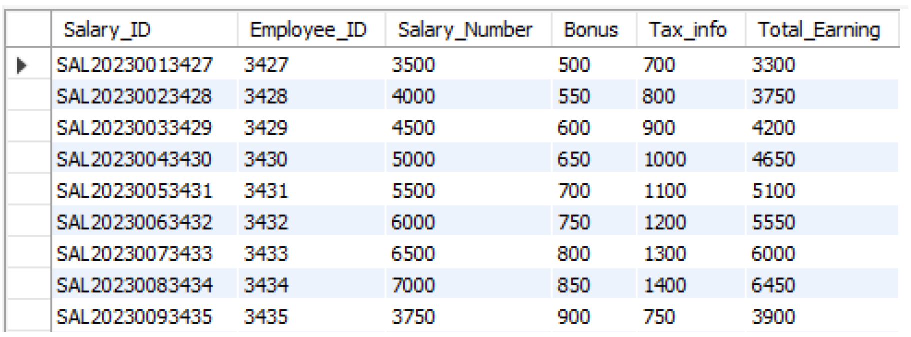


Subscription_ID	User_ID	Payment_Type	Card_Number	Transaction_ID	Subscription_Type
90616604	3020	Credit	4541 4394 4757 7490	TRN202320	Basic
90713579	3006	Debit	8570 4531 5147 9060	TRN202306	Premium
90916317	3018	Debit	7919 2531 5497 1770	TRN202318	Standard
91419994	3019	Debit	5551 1517 4706 3740	TRN202319	Premium
92917525	3004	Debit	7255 6094 2451 2640	TRN202304	Standard
93308143	3007	Debit	7937 0646 9304 4890	TRN202307	Standard
93616488	3013	Credit	7613 1484 7345 3190	TRN202313	Standard
94077295	3012	Debit	4858 0488 2503 4130	TRN202312	Premium
94440532	3002	Credit	8157 7134 3810 6720	TRN202302	Premium
94456205	3016	Credit	5923 4744 9081 3030	TRN202316	Premium
94779132	3005	Credit	4435 2929 6603 5740	TRN202305	Basic
95253191	3017	Credit	6442 2532 3536 2590	TRN202317	Basic

5. Salaries Table

```
CREATE TABLE salaries (
   Salary_ID VARCHAR(20) PRIMARY KEY NOT NULL,
   Employee_ID INT NOT NULL,
   Salary_Number INT NOT NULL,
   Bonus INT NOT NULL,
   Tax_info INT NOT NULL,
   Total_Earning INT NOT NULL,
   FOREIGN KEY (Employee_ID) REFERENCES Employee (Employee_ID)
);
```

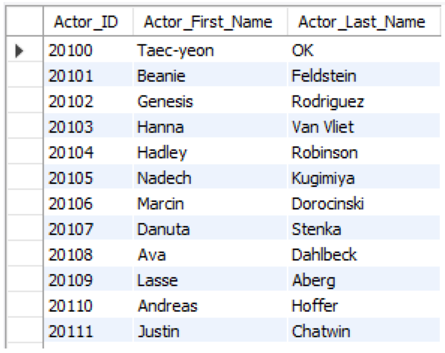
```
INSERT INTO salaries (Salary_ID, Employee_ID, Salary_Number, Bonus, Tax_info,
Total_Earning)

VALUES

('SAL20230013427', 3427, 3500, 500, 700, 3300),

('SAL20230023428', 3428, 4000, 550, 800, 3750),

('SAL20230033429', 3429, 4500, 600, 900, 4200),

('SAL20230043430', 3430, 5000, 650, 1000, 4650),

('SAL20230053431', 3431, 5500, 700, 1100, 5100),

('SAL20230063432', 3432, 6000, 750, 1200, 5550),

('SAL20230073433', 3433, 6500, 800, 1300, 6000),

('SAL20230083434', 3434, 7000, 850, 1400, 6450),

('SAL20230093435', 3435, 3750, 900, 750, 3900),

('SAL20230103436', 3436, 4250, 950, 850, 4350),

('SAL20230113437', 3437, 4750, 525, 950, 4325),

('SAL20230123438', 3438, 5250, 575, 1050, 4775),

('SAL20230133439', 3439, 5750, 625, 1150, 5225),

('SAL20230143440', 3440, 6250, 675, 1250, 5675),

('SAL20230153441', 3441, 6750, 725, 1350, 6125),

('SAL20230163442', 3442, 3900, 775, 780, 3895),

('SAL20230173443', 3443, 4400, 825, 880, 4345),

('SAL20230183444', 3444, 4900, 875, 980, 4795),

('SAL20230193445', 3445, 5400, 925, 1080, 5245),

('SAL20230203446', 3446, 5900, 975, 1180, 5695);
```



Salary_ID	Employee_ID	Salary_Number	Bonus	Tax_info	Total_Earning
SAL20230013427	3427	3500	500	700	3300
SAL20230023428	3428	4000	550	800	3750
SAL20230033429	3429	4500	600	900	4200
SAL20230043430	3430	5000	650	1000	4650
SAL20230053431	3431	5500	700	1100	5100
SAL20230063432	3432	6000	750	1200	5550
SAL20230073433	3433	6500	800	1300	6000
SAL20230083434	3434	7000	850	1400	6450
SAL20230093435	3435	3750	900	750	3900

6. Actor Table

```
Create Table Actor (
Actor_ID Int not null,
Actor_First_Name varchar(22) not null,
Actor_Last_Name varchar(22) not null,
primary key (Actor_ID)
);

INSERT INTO actor (Actor_ID, Actor_First_Name, Actor_Last_Name) VALUES
(20100, 'Taec-yeon', 'OK'),
(20101, 'Beanie', 'Feldstein'),
(20102, 'Genesis', 'Rodriguez'),
(20103, 'Hanna', 'Van Vliet'),
(20104, 'Hadley', 'Robinson'),
(20105, 'Nadech', 'Kugimiya'),
(20106, 'Marcin', 'Dorocinski'),
(20107, 'Danuta', 'Stenka'),
(20108, 'Ava', 'Dahlbeck'),
(20109, 'Lasse', 'Aberg'),
(20110, 'Andreas', 'Hoffer'),
(20111, 'Justin', 'Chatwin'),
(20112, 'Stellan', 'Skarsgard'),
(20113, 'Lasse', 'Aberg'),
(20114, 'Karl', 'Erik'),
(20115, 'Joaquin', 'Phoenix'),
(20116, 'Chiyaan', 'Vikram'),
(20117, 'Ellen', 'Jelinek'),
(20118, 'Micke', 'Andresson'),
(20119, 'Malin', 'Ek');
```



Actor_ID	Actor_First_Name	Actor_Last_Name
20100	Taec-yeon	OK
20101	Beanie	Feldstein
20102	Genesis	Rodriguez
20103	Hanna	Van Vliet
20104	Hadley	Robinson
20105	Nadech	Kugimiya
20106	Marcin	Dorocinski
20107	Danuta	Stenka
20108	Ava	Dahlbeck
20109	Lasse	Aberg
20110	Andreas	Hoffer
20111	Justin	Chatwin

7. Director Table

```
Create Table Director  (
Director_ID  Int not null,
Director_First_Name varchar(22) not null,
Director_last_name varchar(22) not null,
Gender varchar(22) not null,
primary key (Director_ID)
);


INSERT INTO Director (Director_ID, Director_First_Name, Director_Last_Name,
Gender)
VALUES
(4001, 'Joon-hwa', 'Park', 'Male'),
(4002, 'Coky', 'Giedroyc', 'Female'),
(4003, 'Brendan', 'Walsh', 'Male'),
(4004, 'Valerie', 'Bisscheroux', 'Female'),
(4005, 'Amy', 'Poehler', 'Female'),
(4006, 'Mez', 'Tharatorn', 'Male'),
(4007, 'Magdalena', 'Lazakiewicz', 'Female'),
(4008, 'Patryk', 'Vega', 'Male'),
(4009, 'Alf', 'Sjoberg', 'Male'),
(4010, 'Lasse', 'Aberg', 'Male'),
(4011, 'Stephan', 'Apelgren', 'Male'),
```

(4012, 'David', 'S. Goyer', 'Male'),

(4013, 'Hans', 'Alfredson', 'Male'),

(4014, 'Lasse', 'Aberg', 'Male'),

(4015, 'Gosta', 'Werner', 'Male'),

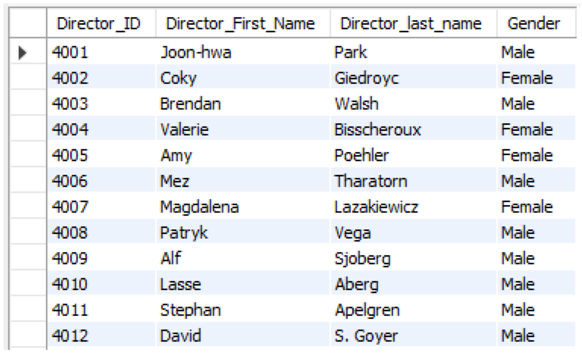
(4016, 'Todd', 'Phillips', 'Male'),

(4017, 'Shankar', 'Shanmugam', 'Male'),

(4018, 'Richard', 'Hobert', 'Male'),

(4019, 'Lasse', 'Hallstrom', 'Male'),

(4020, 'Hans', 'Alfredson', 'Male');



Director_ID	Director_First_Name	Director_last_name	Gender
4001	Joon-hwa	Park	Male
4002	Coky	Giedroyc	Female
4003	Brendan	Walsh	Male
4004	Valerie	Bisscheroux	Female
4005	Amy	Poehler	Female
4006	Mez	Tharatorn	Male
4007	Magdalena	Lazakiewicz	Female
4008	Patryk	Vega	Male
4009	Alf	Sjoberg	Male
4010	Lasse	Aberg	Male
4011	Stephan	Apelgren	Male
4012	David	S. Goyer	Male

8. Region Table

CREATE TABLE Region (

Region_ID INT NOT NULL PRIMARY KEY,

Region_NAME TINYTEXT NOT NULL,

Region_Language TINYTEXT NOT NULL

);

INSERT INTO Region

VALUES

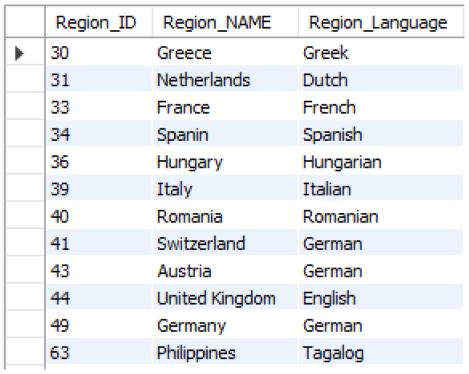
('421','Slovakia','Slovak'),

('359','Bulgaria','Bulgarian'),

('371','Latvia','Latvian'),

('372','Estonia','Estonian'),

```
('354','Iceland','Icelandic'),
('370','Lithuania','Lithuanian'),
('40','Romania','Romanian'),
('44','United Kingdom','English'),
('351','Portugal','Portuguese'),
('353','Ireland','Irish'),
('34','Spanin','Spanish'),
('49','Germany','German'),
('36','Hungary','Hungarian'),
('39','Italy','Italian'),
('43','Austria','German'),
('41','Switzerland','German'),
('63','Philippines','Tagalog'),
('30','Greece','Greek'),
('31','Netherlands','Dutch'),
('33','France','French');
```

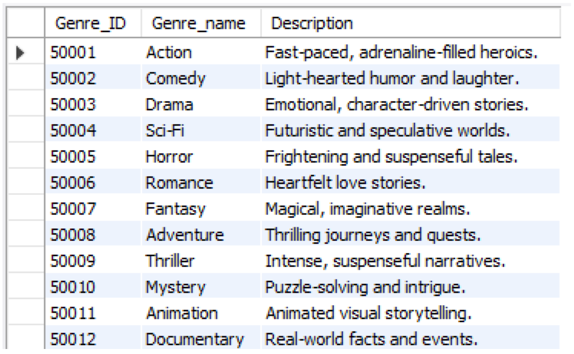


Region_ID	Region_NAME	Region_Language
30	Greece	Greek
31	Netherlands	Dutch
33	France	French
34	Spanin	Spanish
36	Hungary	Hungarian
39	Italy	Italian
40	Romania	Romanian
41	Switzerland	German
43	Austria	German
44	United Kingdom	English
49	Germany	German
63	Philippines	Tagalog

9. Genre Table

```
CREATE TABLE genre (
  Genre_ID INT PRIMARY KEY NOT NULL,
  Genre_name VARCHAR(200) NOT NULL,
  Description TEXT NOT NULL
);
```

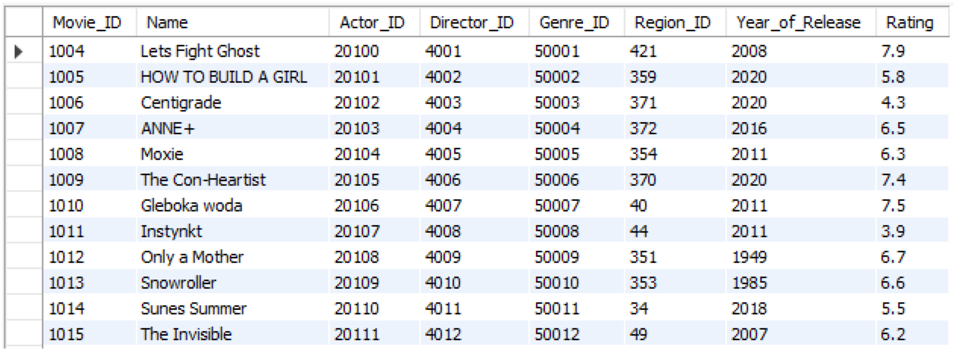
```
INSERT INTO genre (Genre_ID, Genre_name, Description)
VALUES
(50001, 'Action', 'Fast-paced, adrenaline-filled heroics.'),
(50002, 'Comedy', 'Light-hearted humor and laughter.'),
(50003, 'Drama', 'Emotional, character-driven stories.'),
(50004, 'Sci-Fi', 'Futuristic and speculative worlds.'),
(50005, 'Horror', 'Frightening and suspenseful tales.'),
(50006, 'Romance', 'Heartfelt love stories.'),
(50007, 'Fantasy', 'Magical, imaginative realms.'),
(50008, 'Adventure', 'Thrilling journeys and quests.'),
(50009, 'Thriller', 'Intense, suspenseful narratives.'),
(50010, 'Mystery', 'Puzzle-solving and intrigue.'),
(50011, 'Animation', 'Animated visual storytelling.'),
(50012, 'Documentary', 'Real-world facts and events.'),
(50013, 'Crime', 'Criminal activities and investigations.'),
(50014, 'Musical', 'Music-driven storytelling.'),
(50015, 'Historical', 'Stories from the past.'),
(50016, 'Western', 'Wild West adventures.'),
(50017, 'War', 'Historical wartime dramas.'),
(50018, 'Superhero', 'Powerful heroes and villains.'),
(50019, 'Family', 'Entertainment for all ages.'),
(50020, 'Fantasy', 'Mystical and epic adventures.');
```



Genre_ID	Genre_name	Description
50001	Action	Fast-paced, adrenaline-filled heroics.
50002	Comedy	Light-hearted humor and laughter.
50003	Drama	Emotional, character-driven stories.
50004	Sci-Fi	Futuristic and speculative worlds.
50005	Horror	Frightening and suspenseful tales.
50006	Romance	Heartfelt love stories.
50007	Fantasy	Magical, imaginative realms.
50008	Adventure	Thrilling journeys and quests.
50009	Thriller	Intense, suspenseful narratives.
50010	Mystery	Puzzle-solving and intrigue.
50011	Animation	Animated visual storytelling.
50012	Documentary	Real-world facts and events.

10. Movies Table

```
CREATE TABLE Movies (
   Movie_ID INT PRIMARY KEY,
   Name VARCHAR(255),
   Actor_ID INT,
   Director_ID INT,
   Genre_ID INT,
   Region_ID INT,
   Year_of_Release INT,
   Rating DECIMAL(3,1),
   FOREIGN KEY (Actor_ID) REFERENCES Actor(Actor_ID),
   FOREIGN KEY (Director_ID) REFERENCES Director(Director_ID),
   FOREIGN KEY (Genre_ID) REFERENCES Genre(Genre_ID),
   FOREIGN KEY (Region_ID) REFERENCES Region(Region_ID)
);

INSERT INTO movies (Movie_ID, Name, Actor_ID, Director_ID, Genre_ID,
Region_ID, Year_of_Release, Rating)
VALUES
(1004, 'Lets Fight Ghost', 20100, 4001, 50001, 421, 2008, 7.9),
(1005, 'HOW TO BUILD A GIRL', 20101, 4002, 50002, 359, 2020, 5.8),
(1006, 'Centigrade', 20102, 4003, 50003, 371, 2020, 4.3),
(1007, 'ANNE+', 20103, 4004, 50004, 372, 2016, 6.5),
(1008, 'Moxie', 20104, 4005, 50005, 354, 2011, 6.3),
(1009, 'The Con-Heartist', 20105, 4006, 50006, 370, 2020, 7.4),
(1010, 'Gleboka woda', 20106, 4007, 50007, 40, 2011, 7.5),
(1011, 'Instynkt', 20107, 4008, 50008, 44, 2011, 3.9),
(1012, 'Only a Mother', 20108, 4009, 50009, 351, 1949, 6.7),
(1013, 'Snowroller', 20109, 4010, 50010, 353, 1985, 6.6),
(1014, 'Sunes Summer', 20110, 4011, 50011, 34, 2018, 5.5),
(1015, 'The Invisible', 20111, 4012, 50012, 49, 2007, 6.2),
(1016, 'The Simple Minded Murderer', 20112, 4013, 50013, 36, 1985, 7.6),
(1017, 'The Stig-Helmer Story', 20113, 4014, 50014, 39, 2011, 4.5),
(1018, 'To Kill a Child', 20114, 4015, 50015, 43, 2011, 7.7),
(1019, 'Joker', 20115, 4016, 50016, 41, 2019, 8.4),
(1020, 'I', 20116, 4017, 50017, 63, 1999, 6.5),
(1021, 'Harrys Daughters', 20117, 4018, 50018, 30, 2011, 8.1),
(1022, 'Gyllene Tider', 20118, 4019, 50019, 31, 1981, 7.7),
(1023, 'False As Water', 20119, 4020, 50020, 33, 1985, 6.3);
```

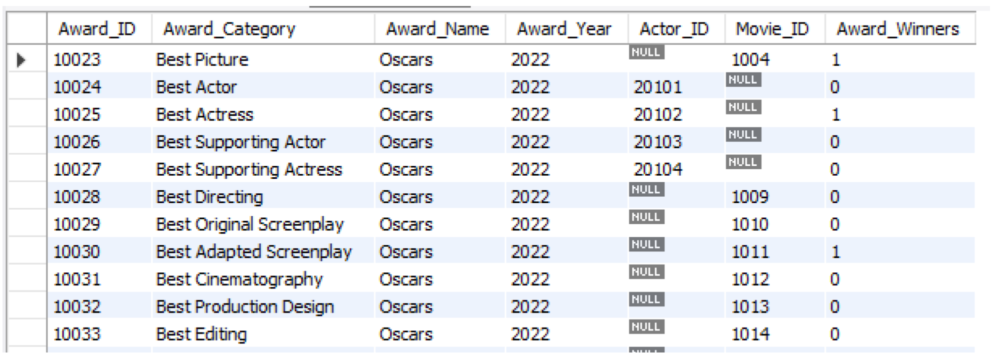


Movie_ID	Name	Actor_ID	Director_ID	Genre_ID	Region_ID	Year_of_Release	Rating
1004	Lets Fight Ghost	20100	4001	50001	421	2008	7.9
1005	HOW TO BUILD A GIRL	20101	4002	50002	359	2020	5.8
1006	Centigrade	20102	4003	50003	371	2020	4.3
1007	ANNE+	20103	4004	50004	372	2016	6.5
1008	Moxie	20104	4005	50005	354	2011	6.3
1009	The Con-Heartist	20105	4006	50006	370	2020	7.4
1010	Gleboka woda	20106	4007	50007	40	2011	7.5
1011	Instynkt	20107	4008	50008	44	2011	3.9
1012	Only a Mother	20108	4009	50009	351	1949	6.7
1013	Snowroller	20109	4010	50010	353	1985	6.6
1014	Sunes Summer	20110	4011	50011	34	2018	5.5
1015	The Invisible	20111	4012	50012	49	2007	6.2

11. Awards Table

```
CREATE TABLE Awards (
    Award_ID INT PRIMARY KEY ,
    Award_Category VARCHAR(255),
    Award_Name VARCHAR(255),
    Award_Year INT,
    Actor_ID INT,
    Movie_ID INT,
    Award_Winners INT,
    FOREIGN KEY (Actor_ID) REFERENCES Actor(Actor_ID),
    FOREIGN KEY (Movie_ID) REFERENCES Movies(Movie_ID)
);
```

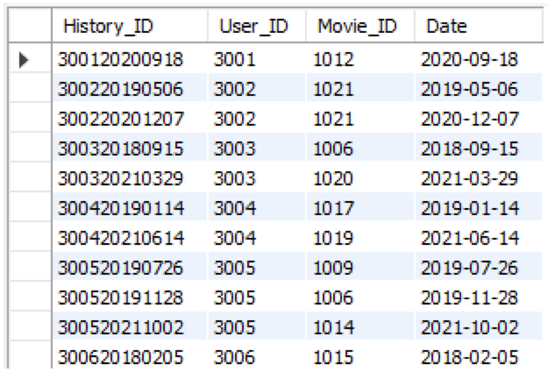
```
INSERT INTO Awards (Award_ID, Award_Category, Award_Name, Award_Year,
Actor_ID, Movie_ID, Award_Winners)
VALUES
    (10023, 'Best Picture', 'Oscars', 2022, NULL, 1004, 1),
    (10024, 'Best Actor', 'Oscars', 2022, 20101, NULL, 0),
    (10025, 'Best Actress', 'Oscars', 2022, 20102, NULL, 1),
    (10026, 'Best Supporting Actor', 'Oscars', 2022, 20103, NULL, 0),
    (10027, 'Best Supporting Actress', 'Oscars', 2022, 20104, NULL, 0),
    (10028, 'Best Directing', 'Oscars', 2022, NULL, 1009, 0),
    (10029, 'Best Original Screenplay', 'Oscars', 2022, NULL, 1010, 0),
    (10030, 'Best Adapted Screenplay', 'Oscars', 2022, NULL, 1011, 1),
    (10031, 'Best Cinematography', 'Oscars', 2022, NULL, 1012, 0),
    (10032, 'Best Production Design', 'Oscars', 2022, NULL, 1013, 0),
    (10033, 'Best Editing', 'Oscars', 2022, NULL, 1014, 0),
    (10034, 'Best Original Score', 'Oscars', 2022, NULL, 1015, 1),
    (10035, 'Best Original Song', 'Oscars', 2022, NULL, 1016, 0),
    (10036, 'Best Costume Design', 'Oscars', 2022, NULL, 1017, 0),
    (10037, 'Best Makeup and Hairstyling', 'Oscars', 2022, NULL, 1018, 0),
    (10038, 'Best Sound Mixing', 'Oscars', 2022, NULL, 1019, 1),
    (10039, 'Best Sound Editing', 'Oscars', 2022, NULL, 1020, 0),
    (10040, 'Best Visual Effects', 'Oscars', 2022, NULL, 1021, 0),
    (10041, 'Best Foreign-Language Film', 'Oscars', 2022, NULL, 1022, 1),
    (10042, 'Best Animated-Featured Short', 'Oscars', 2022, NULL, 1023, 0);
```



Award_ID	Award_Category	Award_Name	Award_Year	Actor_ID	Movie_ID	Award_Winners
10023	Best Picture	Oscars	2022	NULL	1004	1
10024	Best Actor	Oscars	2022	20101	NULL	0
10025	Best Actress	Oscars	2022	20102	NULL	1
10026	Best Supporting Actor	Oscars	2022	20103	NULL	0
10027	Best Supporting Actress	Oscars	2022	20104	NULL	0
10028	Best Directing	Oscars	2022	NULL	1009	0
10029	Best Original Screenplay	Oscars	2022	NULL	1010	0
10030	Best Adapted Screenplay	Oscars	2022	NULL	1011	1
10031	Best Cinematography	Oscars	2022	NULL	1012	0
10032	Best Production Design	Oscars	2022	NULL	1013	0
10033	Best Editing	Oscars	2022	NULL	1014	0

12. Watch History Table

```
CREATE TABLE watch_history (
   History_ID BIGINT NOT Null PRIMARY KEY,
   User_ID INT not null,
   Movie_ID INT not null,
   Date DATE not null,
   FOREIGN KEY (User_ID) REFERENCES Users(user_ID),
FOREIGN KEY (Movie_ID) REFERENCES Movies(Movie_ID)
);

INSERT INTO watch_history (History_ID, User_ID, Movie_ID, Date)
VALUES
(300120200918, 3001, 1012, '2020-09-18'),
(300220201207, 3002, 1021, '2020-12-07'),
(300320210329, 3003, 1020, '2021-03-29'),
(300420210614, 3004, 1019, '2021-06-14'),
(300520211002, 3005, 1014, '2021-10-02'),
(300620220219, 3006, 1022, '2022-02-19'),
(300720220507, 3007, 1007, '2022-05-07'),
(300820220823, 3008, 1005, '2022-08-23'),
(300920221112, 3009, 1008, '2022-11-12'),
(301020230131, 3010, 1011, '2023-01-31'),
(301120201025, 3011, 1010, '2020-10-25'),
(301220210115, 3012, 1004, '2021-01-15'),
(301320210408, 3013, 1005, '2021-04-08'),
(301420210722, 3014, 1005, '2021-07-22'),
(301520211105, 3015, 1006, '2021-11-05'),
(301620220318, 3016, 1023, '2022-03-18'),
(301720220609, 3017, 1017, '2022-06-09'),
(301820220927, 3018, 1007, '2022-09-27'),
(301920221219, 3019, 1006, '2022-12-19'),
(302020230211, 3020, 1018, '2023-02-11'),
(301020170315, 3010, 1019, '2017-03-15'),
(300820170728, 3008, 1005, '2017-07-28'),
(301720171019, 3017, 1020, '2017-10-19'),
(300620180205, 3006, 1015, '2018-02-05'),
(301420180521, 3014, 1012, '2018-05-21'),
(301120180809, 3011, 1008, '2018-08-09'),
(300420190114, 3004, 1017, '2019-01-14'),
(301920190402, 3019, 1010, '2019-04-02'),
(300520190726, 3005, 1009, '2019-07-26'),
(301520191011, 3015, 1004, '2019-10-11'),
(300720170107, 3007, 1005, '2017-01-07'),
(301220170519, 3012, 1014, '2017-05-19'),
(301620170811, 3016, 1007, '2017-08-11'),
(300920180302, 3009, 1013, '2018-03-02'),
(301020180629, 3010, 1019, '2018-06-29'),
(300320180915, 3003, 1006, '2018-09-15'),
(301820190223, 3018, 1018, '2019-02-23'),
(300220190506, 3002, 1021, '2019-05-06'),
(301320190823, 3013, 1016, '2019-08-23'),
(300520191128, 3005, 1006, '2019-11-28');
```

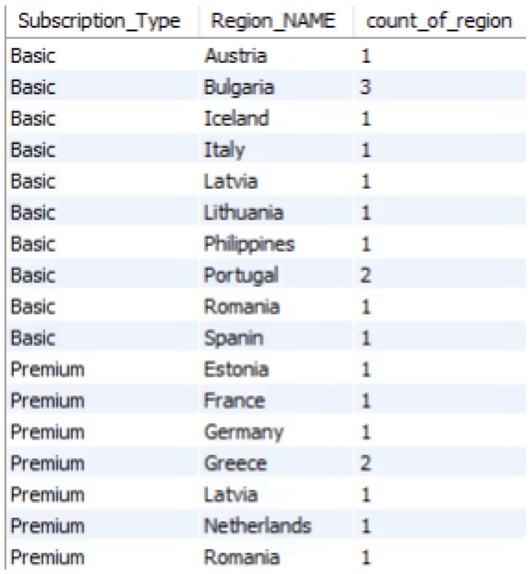


History_ID	User_ID	Movie_ID	Date
300120200918	3001	1012	2020-09-18
300220190506	3002	1021	2019-05-06
300220201207	3002	1021	2020-12-07
300320180915	3003	1006	2018-09-15
300320210329	3003	1020	2021-03-29
300420190114	3004	1017	2019-01-14
300420210614	3004	1019	2021-06-14
300520190726	3005	1009	2019-07-26
300520191128	3005	1006	2019-11-28
300520211002	3005	1014	2021-10-02
300620180205	3006	1015	2018-02-05

PART 4. Ten Advanced Queries for Business Summary and Operation Synthesis

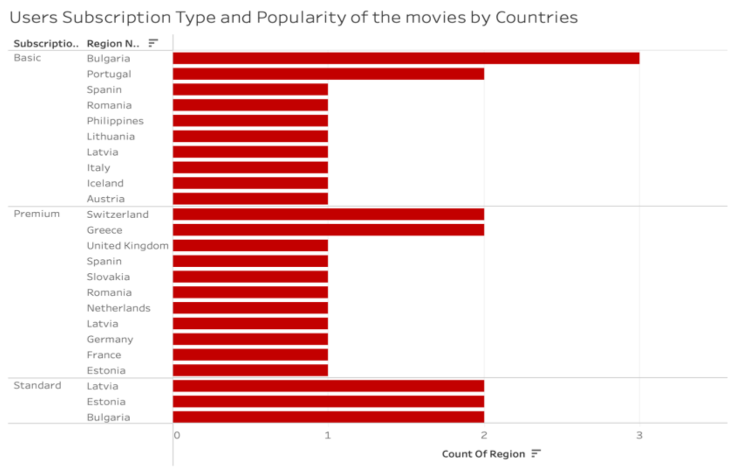
1. **We want to study the relationship between different subscribers and the region of movies they prefer in order to analyze a better marketing strategy for creating movie recommendation lists for different types of users. In addition, we create a view to summarize the subscription type and the count of movie regions based on their watching history.**

```
select age, genre_name,count(genre.Genre_ID)
from users, watch_history,genre, movies
where users.User_id=watch_history.User_ID
and watch_history.Movie_ID=movies.Movie_ID
and movies.Genre_ID=genre.Genre_ID
group by age, genre_name;
```



Subscription_Type	Region_NAME	count_of_region
Basic	Austria	1
Basic	Bulgaria	3
Basic	Iceland	1
Basic	Italy	1
Basic	Latvia	1
Basic	Lithuania	1
Basic	Philippines	1
Basic	Portugal	2
Basic	Romania	1
Basic	Spanin	1
Premium	Estonia	1
Premium	France	1
Premium	Germany	1
Premium	Greece	2
Premium	Latvia	1
Premium	Netherlands	1
Premium	Romania	1

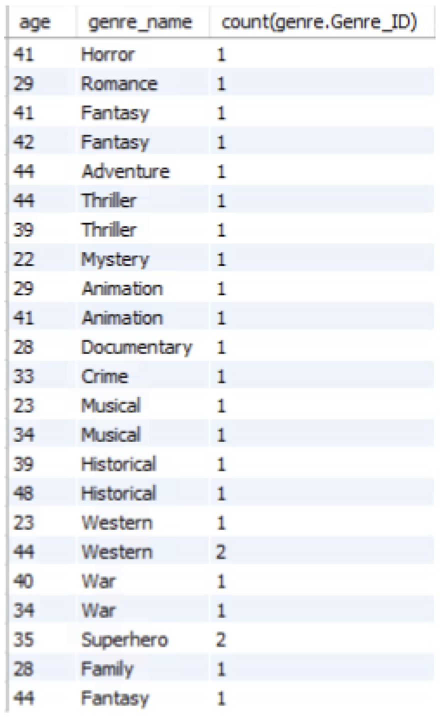
Basic
SUM(Count Of Region)
Standard
SUM(Count Of Region)
Premium
SUM(Count Of Region)



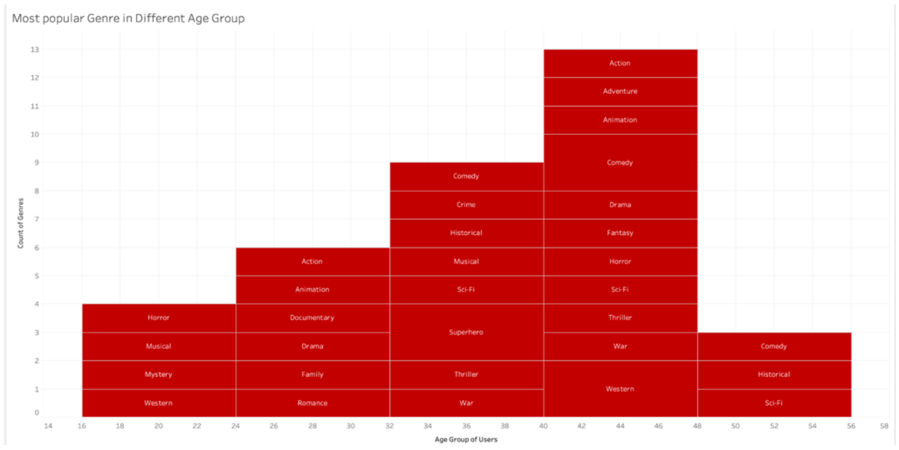
Our Basic subscribers like movies from Bulgaria the most. And Portugal comes as the second. The Premium subscribers prefer movies from Switzerland and Greece. Standard subscribers equally like the movies from Latvia, Estoni, and Bulgaria. We recommend streaming more movies from those top regions to the subscribers based on the subscription types to improve the users' engagement in Netflix.

2. **We want to investigate the popularity of different movie genres by age group. Summarizing user age and their preferred genre also improves our marketing strategy's precision.**

```
select age, genre_name,count(genre.Genre_ID)
from users, watch_history,genre, movies
where users.User_id=watch_history.User_ID
and watch_history.Movie_ID=movies.Movie_ID
and movies.Genre_ID=genre.Genre_ID
group by age, genre_name;
```



age	genre_name	count(genre.Genre_ID)
41	Horror	1
29	Romance	1
41	Fantasy	1
42	Fantasy	1
44	Adventure	1
44	Thriller	1
39	Thriller	1
22	Mystery	1
29	Animation	1
41	Animation	1
28	Documentary	1
33	Crime	1
23	Musical	1
34	Musical	1
39	Historical	1
48	Historical	1
23	Western	1
44	Western	2
40	War	1
34	War	1
35	Superhero	2
28	Family	1
44	Fantasy	1



The age of our users recorded in the database ranges from 16 to 56 years old. We divide them into 5 different groups to analyze what genres are the most popular in each age group and why certain age groups have fewer genres preferred than others.

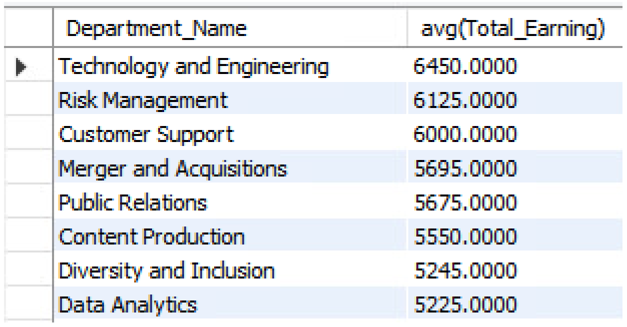
Young people in the age group 16-23 love horror, musical mystery, and western movies. Age group 24-31 loves Action, animation, documentaries, drama, family, and romance movies. The age group 32-40 has a wider spread of genres. Among all those genres, they love superhero movies the most. It might be attributed to the flourishing of Marvel animation at a young age.

The age group 40-47 has the most watching histories, and they have a significant preference for comedy and Western movies. Those users have the highest level of user engagement on Netflix. They spend more spare time watching movies compared to other groups. It's probably because they have more disposable time as they have a higher income level than the younger generations.

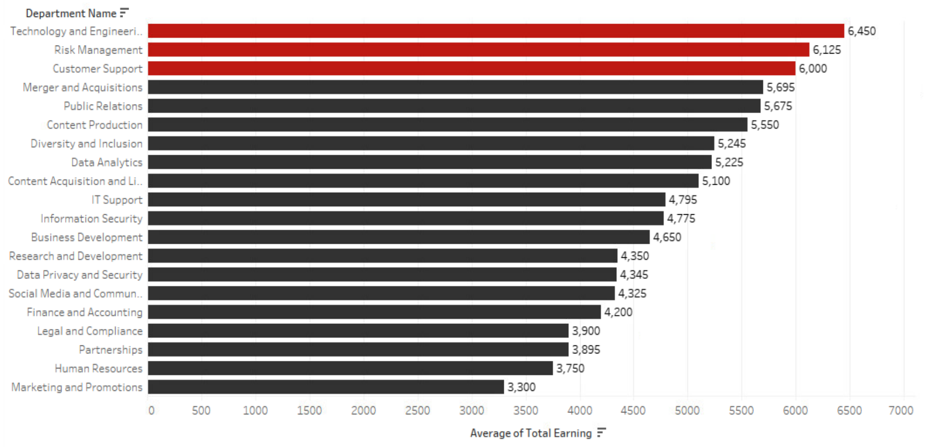
The age group above 48 tends to have more constant genres to watch: comedy, Historical, and Sci-Fi. Netflix should invest in promoting different varieties of movies to users above 48 years old and provide acceptable help to transform users' habits from traditional TV to streaming media since they have a great potential to spend more time with Netflix, considering the age group has a smaller working population.

3. Investigate average salary across different departments

```
SELECT department.Department_Name, avg(Total_Earning)
FROM department, salaries, employee
WHERE employee.Department_ID = department.Department_ID
AND employee.Employee_ID = salaries.Employee_ID
GROUP BY department.Department_Name
ORDER BY AVG(Total_Earning) DESC;
```



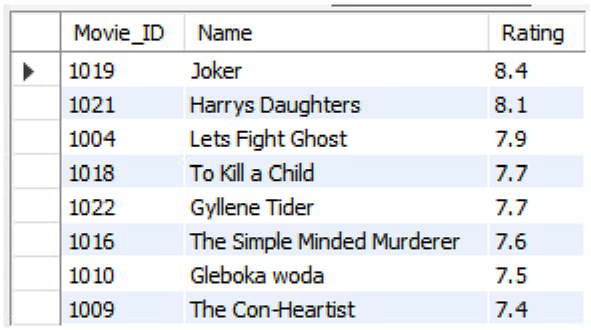
Department_Name	avg(Total_Earning)
Technology and Engineering	6450.0000
Risk Management	6125.0000
Customer Support	6000.0000
Merger and Acquisitions	5695.0000
Public Relations	5675.0000
Content Production	5550.0000
Diversity and Inclusion	5245.0000
Data Analytics	5225.0000



The analysis of average salaries across departments reveals a positive trend, with Techonology and Engineering, Risk Management, and Customer Support all boasting average salaries surpassing 6000. This underscores the organization's commitment to recognizing the value of technical expertise, effective risk management, and exceptional customer support. The lower average salaries observed in Partnerships, Human Resrouces, and Marketing and Promotion departments suggest areas for potential improvement in compensation structures. To address these lower averages, a comprehensive review of market benchmarks, employee satisfaction, and the strategic importance of these department is recommended. Frequent reviews and adjustment to ensure competitiveness with industry standards are important for maintaining efficiency in workforce.

4. **Design a SQL procedure to efficiently retrieve a list of movies with ratings equal to or higher than 7.0. We want to discuss strategic advantages for Netflix in terms of content recommendation, user engagement, and overall platform performance. Create a list procedure for the movies that have at least 7.0 ratings**

```
delimiter //
CREATE PROCEDURE GetTopRatingMovies()
BEGIN
SELECT Movie_ID, Name, Rating
FROM movies
WHERE Rating >= 7
ORDER BY Rating DESC;
END;
//


CALL GetTopRatingMovies()
```

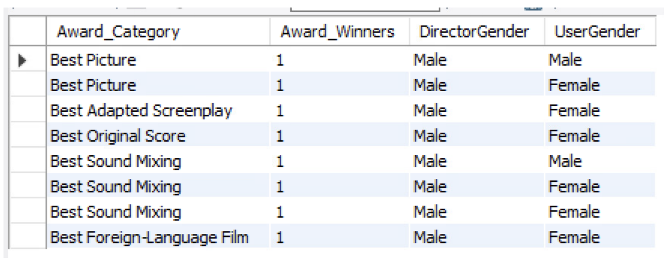


Movie_ID	Name	Rating
1019	Joker	8.4
1021	Harrys Daughters	8.1
1004	Lets Fight Ghost	7.9
1018	To Kill a Child	7.7
1022	Gyllene Tider	7.7
1016	The Simple Minded Murderer	7.6
1010	Gleboka woda	7.5
1009	The Con-Heartist	7.4

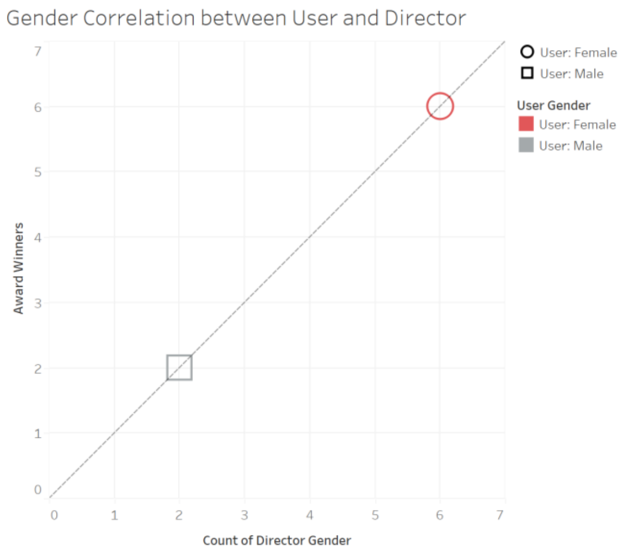
This procedure serves as a foundational tool for content curation, helping in personalizing and high-quality recommending to the users. By efficiently filtering movies based on their ratings, hte platform can enhance user satisfaction, engaging with the customers, and encouraging longer watching sessions.

5. **Do certain demographics of directors appeal to certain demographics of users?**

```
SELECT Award_Category, Award_Winners, director.Gender AS DirectorGender,
users.gender AS UserGender
FROM Awards
INNER JOIN Movies ON Awards.Movie_ID = Movies.Movie_ID
INNER JOIN Watch_History ON Movies.Movie_ID = Watch_History.Movie_ID
INNER JOIN Director ON director.Director_ID = movies.Director_ID
INNER JOIN Users ON Users.User_ID = Watch_History.User_ID
WHERE Award_Winners = 1
```



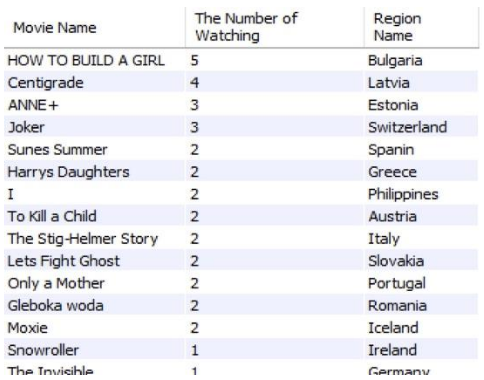
Award_Category	Award_Winners	DirectorGender	UserGender
Best Picture	1	Male	Male
Best Picture	1	Male	Female
Best Adapted Screenplay	1	Male	Female
Best Original Score	1	Male	Female
Best Sound Mixing	1	Male	Male
Best Sound Mixing	1	Male	Female
Best Sound Mixing	1	Male	Female
Best Foreign-Language Film	1	Male	Female



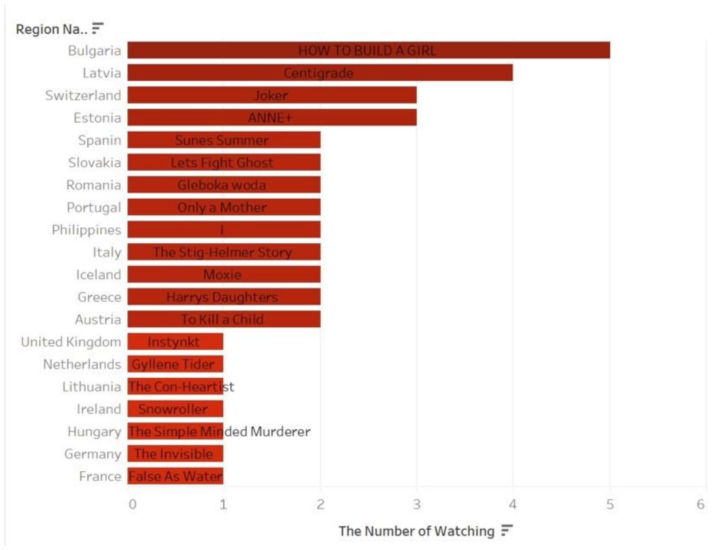
It seems that certain demographics of directors do not have an appeal to certain demographics of users. Out of the movies offered by Netflix, we have used a sample of award-winning movies due to these movies being nationally and internationally proclaimed works of film in their respective areas. We have found little correlation that female users prefer female directors and male users prefer male directors.
We can conclude that Netflix as a business does not need to take into consideration the gender of director or user when recommending movies, especially award-winning movies.
We have further hypothesized that other factors such as the gender of the main actor, or the sample size of award-winning movies might have swayed in the preference, but this would need more data and research to confirm these elements.

6. The number of movies watching in each Region.

```
select Name as 'Movie Name' , count(Date) as 'The Number of Watching',
Region_NAME as 'Region Name'
from watch_history, movies, region
where watch_history.Movie_ID = movies.Movie_ID
and movies.Region_ID = region.Region_ID
group by movies.Movie_ID
order by count(Date) desc;
```



Movie Name	The Number of Watching	Region Name
HOW TO BUILD A GIRL	5	Bulgaria
Centigrade	4	Latvia
ANNE+	3	Estonia
Joker	3	Switzerland
Sunes Summer	2	Spanin
Harrys Daughters	2	Greece
I	2	Philippines
To Kill a Child	2	Austria
The Stig-Helmer Story	2	Italy
Lets Fight Ghost	2	Slovakia
Only a Mother	2	Portugal
Gleboka woda	2	Romania
Moxie	2	Iceland
Snowroller	1	Ireland
The Invisible	1	Germany

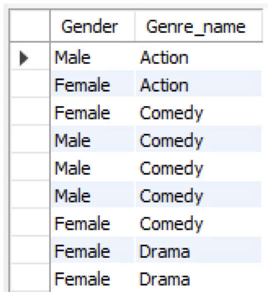


It shows some difference between the number of people watching in the first four regions compared to the rest of the regions. Bulgaria has the highest number which is 5. Second is Latvia has 4 views. Then, Estonia and Switzerland have 3 views. Also, the rest of the nine regions have two views such as Spain, Australia and Romania. And regions like the United Kingdom and Germany have only 1 view along with 5 other regions. This graph can provide Netflix with insights regarding their investment decisions. It allows them to know what type of movies are the people's favorite in a specific region. For example, Bulgaria prefers comedy movies such as How to build a girl, thus Netflix can invest in comedy movies or make a new season for How to Build a Girl. Such an insight can make a big difference when targeting a specific audience.

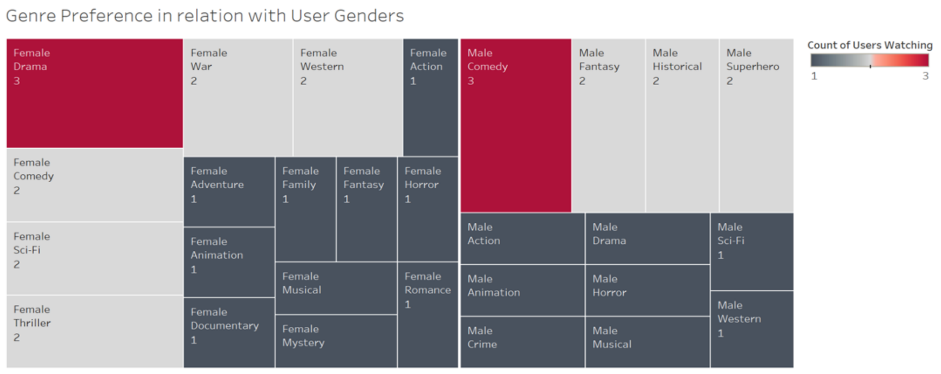
7. **What is the Relation between Gender of User and Genre of Movie**
   Create a view to present a more concise table of the user's gender in relation to the user id

```
CREATE VIEW User_Gender AS
SELECT User_ID, Gender
FROM users;
```

```
SELECT Gender, Genre_name
FROM user_gender
INNER JOIN Watch_History ON user_gender.User_ID = Watch_History.User_ID
INNER JOIN Movies ON Movies.Movie_ID = Watch_History.Movie_ID
INNER JOIN Genre ON Movies.Genre_ID = Genre.Genre_ID;
```



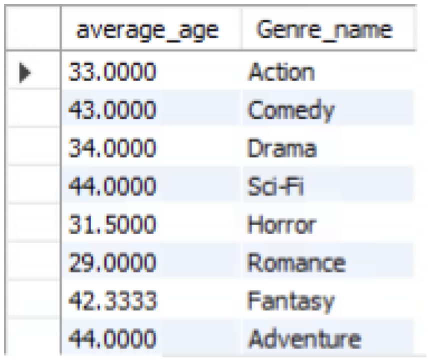
Gender	Genre_name
Male	Action
Female	Action
Female	Comedy
Male	Comedy
Male	Comedy
Male	Comedy
Female	Comedy
Female	Drama
Female	Drama



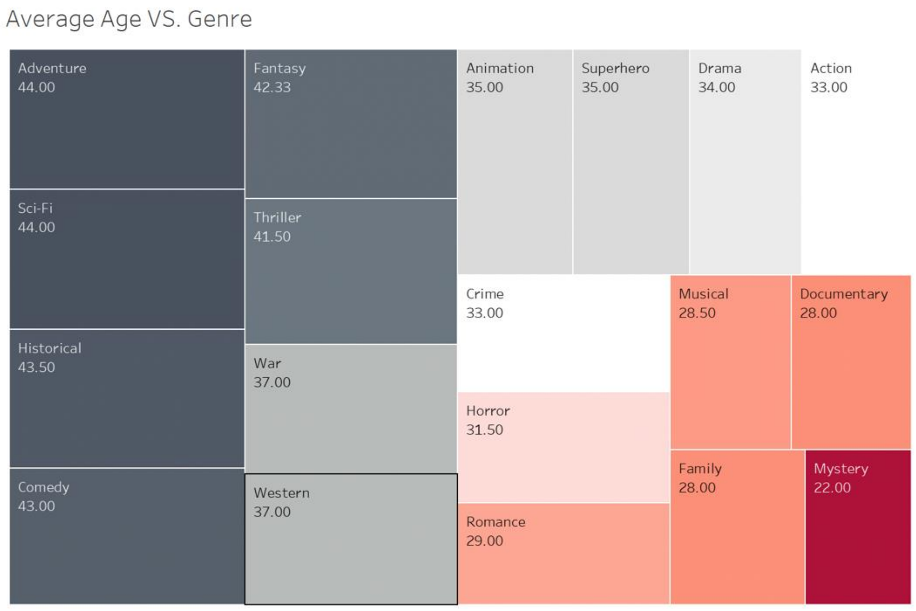
It seems that different genres appeal to certain demographics of users. We can see that female users enjoy Drama the most while Males enjoy Comedy the most. However when we consider movie genres that do not appeal to both genders of the user, we can see Action, Animation, and Musical, and some genres don't even have a userbase watching them.
Netflix as a business might consider investing in more movies that fall in the genre of Drama and Comedy, while cutting down on streaming movies that are in the genre of Action, Animation, and Musical as well as other genres that do not have a very active user watch base.

8. Relationship between average age of users who watch movies of a specific genre.

```
SELECT AVG(YEAR(CURRENT_DATE) - YEAR(u.Date_of_Birth)) AS average_age,
g.Genre_name
FROM Users u
JOIN Watch_History wh ON u.User_ID = wh.User_ID
JOIN Movies m ON wh.Movie_ID = m.Movie_ID
JOIN Genre g ON m.Genre_ID = g.Genre_ID
GROUP BY g.Genre_name;
```



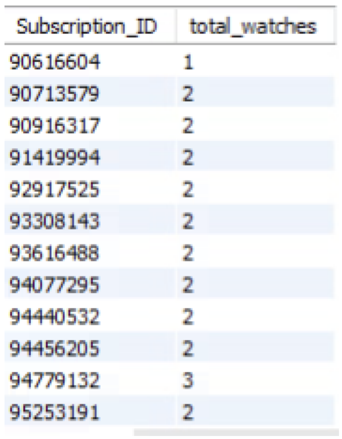
	average_age	Genre_name
▶	33.0000	Action
	43.0000	Comedy
	34.0000	Drama
	44.0000	Sci-Fi
	31.5000	Horror
	29.0000	Romance
	42.3333	Fantasy
	44.0000	Adventure



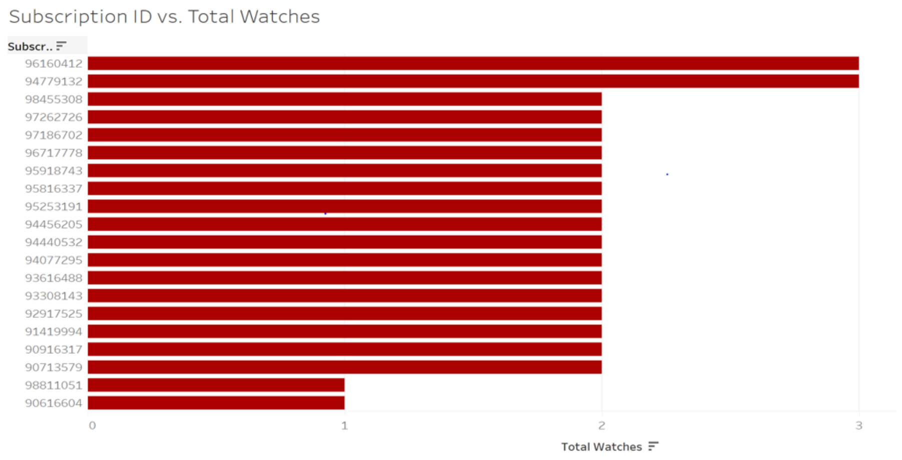
This query calculates the average age of users watching movies of different genres. It provides insights into the age demographics of viewership for each genre, helping to inform marketing strategies and content curation. For instance, if the average age for a genre like 'Animation' is lower, it might suggest that younger viewers prefer this genre, and marketing efforts can be directed accordingly.

9. Summary of the total number of movies watched per subscription ID

```
CREATE VIEW SubscriptionMovieWatchSummary AS
SELECT s.Subscription_ID, COUNT(wh.History_ID) AS total_watches
FROM Subscription s
JOIN Users subscriptionmoviewatchsummarycu ON s.User_ID = u.User_ID
JOIN Watch_History wh ON u.User_ID = wh.User_ID
GROUP BY s.Subscription_ID;
```



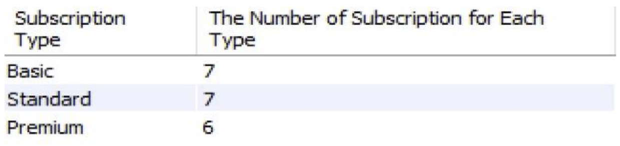
Subscription_ID	total_watches
90616604	1
90713579	2
90916317	2
91419994	2
92917525	2
93308143	2
93616488	2
94077295	2
94440532	2
94456205	2
94779132	3
95253191	2



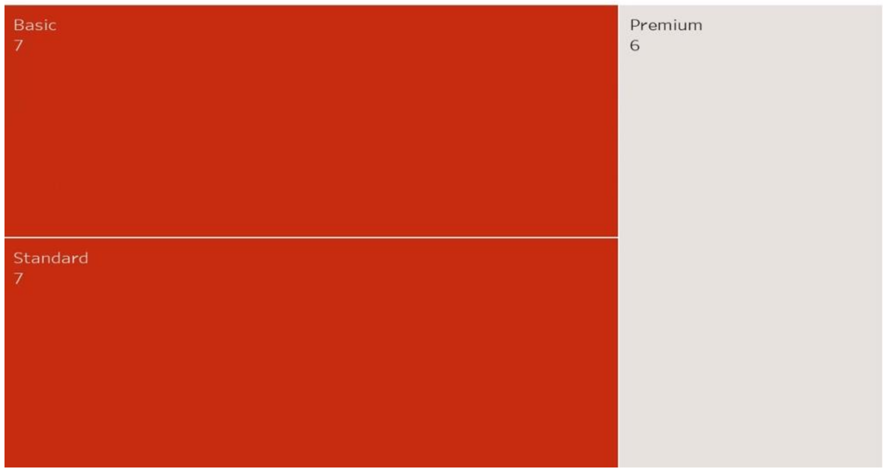
This view creation query aggregates the total number of movies watched per subscription, giving a clear metric of engagement at the subscription level. It can help identify which subscription plans see the most activity and if certain plans are more conducive to higher viewership. This data could be crucial for adjusting subscription plans, pricing strategies, and understanding the return on investment for each plan type.

10. Summary on the most popular subscription Type

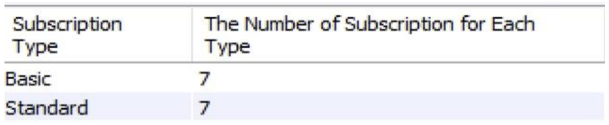
```
select Subscription_Type as 'Subscription Type', count(Subscription_Type) ' The Number of Subscription for Each Type'
from subscription
group by Subscription_Type
order by count(Subscription_Type) desc;
```



Subscription Type	The Number of Subscription for Each Type
Basic	7
Standard	7
Premium	6



```
delimiter //
CREATE PROCEDURE TheMostSubscription()
BEGIN
SELECT Subscription_Type as ' Subscription Type' , count(Subscription_Type) as ' The
Number of Subscription for Each Type'
from subscription
group by Subscription_Type
having count(Subscription_Type) >= 7
order by count(Subscription_Type) desc;


END;
// CALL TheMostSubscription()
```



Subscription Type	The Number of Subscription for Each Type
Basic	7
Standard	7

It shows there are three types of subscriptions in Netflix. Basic and standard subscriptions have 7 subscribers and premium with 6 subscribers. This gives a good indication that Netflix subscription categories are well-established and suits different people. However, if Netflix's goal is to increase their revenue, a suggestion would be to include more features into the premium subscription or to find a price point between Standard and Premium subscriptions. Both recommendations are to encourage subscribers to upgrade to the Premium type.

References

https://www.kaggle.com/datasets/tawfikelmetwally/employee-dataset
https://www.kaggle.com/datasets/arnavsmayan/netflix-userbase-dataset
https://www.kaggle.com/datasets/ashishgup/netflix-rotten-tomatoes-metacritic-imdb
https://www.kaggle.com/datasets/ashishgup/netflix-rotten-tomatoes-metacritic-imdb

https://worldpopulationreview.com/country-rankings/what-country-has-the-best-netflix