Laura Yuan
Max Wu
Yoga Ramachandran

Final Report - Group 7
Database Management Systems - Fundamentals of SQL

Week 1

Netflix

1. **Write a description of the business. What items do they sell?**

Netflix is the world's top online streaming platform with over 193 million subscribers in more than 190 countries. The Company primarily gains revenue from monthly subscriptions through its users. Netflix is also a production company and is currently offering approximately 37 Netflix original productions as of 2020, on its online streaming platforms. The Company's main expenses are for licensing the movies/TV shows offered on the platform as well as marketing costs. Users who use Netflix will have the options to choose between three plans offered: Basic at $8.99, Standard at $13.99, and Premium at $17.99. Different plans offer different quality of viewings (Basic, HD, and Ultra-HD), different limits of the number of devices that can be viewed at the same time (1-4 devices), and different limits on the number of phones or tablets you can have downloads on (1-4 devices).

2. **List the data that can be collected which may be useful.**

1. **Customer billing**- BillingID (primary key), Type of Plan, CustomerID, Start date, End date(for inactive customers)
2. **Customer complaint**s - ComplaintID (primary key), CustomerID, Date, Complaint description, Status
3. **Customer** - CustomerID (primary key), First Name, Last Name, E-mail, Phone, Credit Card Name, Credit Card Number, Credit Card Bank, Status(Active/Not active), Country, PlanID
4. **Profile** - ProfileID, CustomerID, Profile name, Parental restriction(Yes/No)
5. **Content** - ContentID (primary key), Name, Netflix Original(Yes/No), Genre, Type, Release date on Netflix, Duration, CastID, Licensing cost, Duration of licensing, Number of seasons, Number of episodes, Production Company
6. **Customer Viewing History** - ProfileID(linked), ContentID (linked), View time(DateTime), Duration
7. **Plans** - PlanID, Type, Rate, Screen Limit, Resolution,Screen download limit
8. **Cast** - CastID, Director, Actor, Actress

3. **List three business questions you may want to find out from the data. These business questions help you make strategic decisions to increase profit**.

1. How can we retain customers? Why are they choosing Netflix?
2. Are customers willing to pay higher prices for more material or higher resolution?
3. How many customers have left Netflix after raising complaints?

Week 2

Create Tables and populate data from Excel. Create at least 2 denormalized tables and at least 3 normalized tables. State which are normalized and which are denormalized.

1. **Creation of 'Customer' Table (Denormalized):**

```
CREATE TABLE Customer (
CustomerID VARCHAR(20) PRIMARY KEY,
First_Name VARCHAR(50) NOT NULL,
Last_Name VARCHAR(50) NOT NULL,
Email VARCHAR(30) NOT NULL,
Phone BIGINT NOT NULL,
Credit_Card_Name VARCHAR(30) NOT NULL,
Credit_Card_Bank VARCHAR(20) NOT NULL,
Credit_Card_Number bigint NOT NULL,
Customer_Status VARCHAR(10),
PlanID VARCHAR(5),
Country VARCHAR(20),
Complaint_date VARCHAR(50),
Complaint_description VARCHAR(1000),
Complaint_Status VARCHAR(20));
```

Field	Type	Null	Key
CustomerID	varchar(20)	NO	PRI
First_Name	varchar(50)	NO	
Last_Name	varchar(50)	NO	
Email	varchar(30)	NO	
Phone	bigint	NO	
Credit_Card_Name	varchar(30)	NO	
Credit_Card_Bank	varchar(20)	NO	
Credit_Card_Number	bigint	NO	
Customer_Status	varchar(10)	YES	
PlanID	varchar(5)	YES	
Country	varchar(20)	YES	
Complaint_date	varchar(50)	YES	
Complaint_description	varchar(1…	YES	
Complaint_Status	varchar(20)	YES	

An example of an INSERT code is as below:
INSERT INTO CUSTOMER VALUES
('C00001','Alby','Carter','abcarter@gmail.com','408198134','Alby Carter','CHASE','1225877768','Not Active','B','USA','2017-07-01 14:00:00,2019-06-01 14:00:00','Resolution is not good, Every time I search a movie I have to delete letter by letter instead of clearing search.. if tech team can include Clear search feature it would be simply great.. especially when we are at holiday season and binge watching','Open,Open');

CustomerID	First_Name	Last_Name	Email	Phone	Credit_Card_Name	Credit_Card_Bank	Credit_Card_Number	Customer_Status	PlanID	Country
C00001	Alby	Carter	abcarter@gmail.com	408198134	Alby Carter	CHASE	1225877768	Not Active	B	USA
C00002	Damien	Dexter	dexter@gmail.com	408108104	Damien Dexter	Bank of America	4381774297	Not Active	S	UK
C00003	Carl	Christy	ccool@gmail.com	512112678	Carl Christy	CHASE	3658811670	Active	S	Australia
C00004	David	Warner	warner@gmail.com	513102674	David Warner	Wells Fargo	6436142279	Not Active	P	USA
C00005	Steven	Hughes	steven01@gmail.com	513102122	Steven Hughes	Capital One	950623186	Active	P	USA
C00006	Emma	Wang	emmaw@gmail.com	408312134	Emma Wang	Credit One	9930775295	Active	B	USA
C00007	Mia	George	mia@gmail.com	408145678	Mia George	Bank of America	2990951395	Not Active	P	UK
C00008	Charlotte	Harper	charper@gmail.com	408105678	Charlotte Harper	Credit One	3303303915	Not Active	S	Canada
C00009	Victoria	Stevenson	vstevenson@gmail.com	408112109	Victoria Stevenson	Capital One	5673804400	Active	S	Australia
C00010	Ellie	Golding	ellie@yahoo.com	512134567	Ellie Golding	CHASE	2748994625	Active	P	Canada
C00011	Olive	Taylor	otaylor@gmail.com	512134569	Olive Taylor	Bank of America	2148994620	Active	B	USA

2. **Creation of 'Content' Table (Denormalized):**

```
CREATE TABLE Content(
ContentID       VARCHAR(50),
Title VARCHAR(50) NOT NULL,
CastID VARCHAR(50),
Netflix_Original CHAR(10) NOT NULL,
Director VARCHAR(50) NOT NULL,
Actor   VARCHAR(50),
ActressVARCHAR(50),
Genre  VARCHAR(20) NOT NULL,
Video_Type    VARCHAR(50) NOT NULL,
Release_Date DATETIME NOT NULL,
Duration SMALLINT,
No_of_Seasons SMALLINT,
No_of_Episodes          SMALLINT,
Production_Company VARCHAR(50) NOT NULL,
Licensing_Cost          BIGINT,
Licensing_Years SMALLINT,
CONSTRAINT PKContentID PRIMARY KEY (ContentID));
```

Column	Type	Nullable	Indexes
ContentID	varchar(50)	NO	PRIMARY
Title	varchar(50)	NO	
CastID	varchar(50)	YES	FKCastID
Netflix_Original	char(10)	NO	
Genre	varchar(20)	NO	
Video_Type	varchar(50)	NO	
Release_Date	datetime	NO	
Duration	smallint	YES	
No_of_Seasons	smallint	YES	
No_of_Episodes	smallint	YES	
Production_Company	varchar(50)	NO	
Licensing_Cost	bigint	YES	
Licensing_Years	smallint	YES	

An example of the INSERT code is as below:
INSERT INTO Content VALUES ('00000M1','Dont Listen','C01','Yes','Angel Gomez Hernandez','Rodolfo Sancho','Ana Fernandez','Horror','Movie','2015-01-02 00:00:00',98,NULL,NULL,'Netflix',0,0);

ContentID	Title	CastID	Netflix_Original	Genre	Video_Type	Release_Date	Duration	No_of_Seasons	No_of_Episodes	Production_Company	Licensing_Cost	Licensing_Years
00000M2	Disclosure	C05	No	Documentary	Movie	2020-01-27 00:00:00	120	NULL	NULL	Field of Vision Bow	150000	6
00000M3	Wildlife	C06	No	Drama	Movie	2018-10-18 00:00:00	115	NULL	NULL	LMC Productions	200000	3
00000M4	On My Skin	C07	No	True Crime	Movie	2018-09-12 00:00:00	110	NULL	NULL	ABC Network	60000	5
00000M5	Moonlight	C10	No	Drama	Movie	2017-02-16 00:00:00	120	NULL	NULL	Plan B Entertainment	0	0
00000M6	Christmas Eve	C11	No	Holiday	Movie	2014-04-26 00:00:00	113	NULL	NULL	ABC Network	480000	9
00000M7	Aliens	C13	No	Drama	Movie	2018-11-11 00:00:00	90	NULL	NULL	Waterworks	145000	2
00000M8	Luck in the night	C16	No	Romance	Movie	2020-08-27 00:00:00	105	NULL	NULL	LBB Vision	90000	5
00000M9	Love itself	C19	Yes	Romance	Movie	2020-01-01 00:00:00	86	NULL	NULL	Netflix	0	0
00000T1	Bonnie and Clyde	C02	No	Romance	TV Show	2017-05-03 00:00:00	NULL	3	24	ABC Network	80000	4
00000T10	Space Adventure	C18	No	Documentary	TV Show	2019-10-20 00:00:00	NULL	3	20	Invision	338000	3

3. **Creation of 'Customer Profiles' Table (Normalized)**:
   CREATE TABLE Customer_Profiles (
   ProfileID VARCHAR(50),
   CustomerID VARCHAR(50),
   Profile_name VARCHAR(20) NOT NULL,
   Parental_restriction CHAR(5) NOT NULL,
   CONSTRAINT PKProfileID PRIMARY KEY (ProfileID));

   Referencing CustomerID as FK in Customer_Profiles:
   ALTER TABLE Customer_Profiles
   ADD FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID);

Column	Type	Nullable	Indexes
ProfileID	varchar(50)	NO	PRIMARY
CustomerID	varchar(50)	YES	CustomerID
Profile_name	varchar(20)	NO	
Parental_restriction	char(5)	NO	

An example of the INSERT code is as below:
INSERT INTO Customer_Profiles VALUES ('P00001','C00001','Albiecutie','No');

ProfileID	CustomerID	Profile_name	Parental_restriction
P00001	C00001	Albiecutie	No
P00002	C00001	Angel65	Yes
P00003	C00002	Damy	No
P00004	C00003	Christy	No
P00005	C00004	David	Yes
P00006	C00002	Blackpuff	No
P00007	C00006	Emma	No
P00008	C00004	D_unicorn	No
P00009	C00005	Steven	Yes
P00010	C00010	Ellie	No
P00011	C00005	Stevestar	Yes
P00012	C00007	Rockstar	Yes
P00013	C00008	Eckart	No
P00014	C00009	Rockie	No
P00015	C00003	Esther	No
P00016	C00010	Adrienne	No
P00017	C00008	Grandpas	No
P00018	C00009	Kids	Yes
P00019	C00002	Adams	No

4. **Creation of 'Plans' Table (Normalized):**

```
CREATE TABLE Plans(
PlanID VARCHAR(50),
Plan_Type      CHAR(10) NOT NULL,
Rate    FLOAT(4,2) NOT NULL,
Resolution      CHAR(10) NOT NULL,
Screen_Limit SMALLINT NOT NULL,
Screen_Download_Limit SMALLINT NOT NULL,
CONSTRAINT PKPlanID PRIMARY KEY (PlanID));
```

Column	Type	Nullable	Indexes
PlanID	varchar(50)	NO	PRIMARY
Plan_Type	char(10)	NO	
Rate	float(4,2)	NO	
Resolution	char(10)	NO	
Screen_Limit	smallint	NO	
Screen_Download_Limit	smallint	NO	

An example of the INSERT code is as below:
INSERT INTO Plans VALUES ('B','Basic',8.99,'Regular',1,1);

PlanID	Plan_Type	Rate	Resolution	Screen_Limit	Screen_Download_Limit
B	Basic	8.99	Regular	1	1
P	Premium	17.99	Ultra HD	4	4
S	Standard	13.99	HD	2	2

5. **Creation of 'Customer Viewing History'(Normalized)**:
CREATE TABLE Customer_Viewing_History (
ProfileID VARCHAR(50),
ContentID VARCHAR(50),
View_time DATETIME NOT NULL,
Duration SMALLINT);

Column	Type	Nullable	Indexes
ProfileID	varchar(50)	YES	
ContentID	varchar(50)	YES	
View_time	datetime	NO	
Duration	smallint	YES	

An example of the INSERT code is as below:
INSERT INTO Customer_Viewing_History VALUES ('P00001','00000M1','2017-12-31 14:23:02',98);

ProfileID	ContentID	View_time	Duration
P00001	00000M1	2017-12-31 14:23:02	98
P00002	00000T1	2017-06-23 23:00:32	240
P00003	00000T2	2020-01-02 07:39:20	200
P00004	00000T3	2019-07-08 12:32:13	105
P00005	00000M2	2018-03-03 20:56:05	100
P00006	00000M3	2015-04-14 18:09:11	60
P00007	00000M4	2019-10-10 08:44:00	35
P00008	00000T4	2020-02-03 10:23:50	145
P00009	00000T5	2016-09-09 21:20:06	360
P00010	00000M5	2017-05-05 09:09:10	80
P00003	00000M6	2019-02-22 14:22:55	91
P00006	00000T6	2016-08-23 22:23:34	3
P00010	00000M7	2017-09-30 08:24:07	488
P00009	00000T7	2019-10-11 23:20:02	315
P00011	00000T8	2017-04-02 04:23:56	90
P00019	00000M8	2018-09-09 14:22:00	364
P00020	00000T9	2020-01-23 16:15:24	100
P00016	00000T10	2017-12-01 11:10:03	39
P00009	00000M8	2016-10-10 13:03:07	419
P00017	00000M10	2018-02-02 04:23:10	45

6. **Creation of 'Customer Billing' (Normalized):**

```
CREATE TABLE Customer_Billing(
BillingID VARCHAR(20),
Type_of_Plan CHAR(5),
CustomerID VARCHAR(20),
Start_Date DATETIME,
End_Date DATETIME,
CONSTRAINT PKBillingID PRIMARY KEY (BillingID));
```

Referencing Type_of_Plan as FK in Customer_Billing:

```
ALTER TABLE Customer_Billing
ADD FOREIGN KEY (Type_of_Plan) REFERENCES Plans(PlanID);
```

Column	Type	Nullable	Indexes
BillingID	varchar(20)	NO	PRIMARY
Type_of_Plan	char(5)	YES	Type_of_Plan
CustomerID	varchar(20)	YES	
Start_Date	datetime	YES	
End_Date	datetime	YES	

An example of the INSERT code is as below:
INSERT INTO Customer_Billing VALUES ('B001','B','C00001','2017-06-01 14:00:00','2020-06-02 14:00:00');

BillingID	Type_of_Plan	CustomerID	Start_Date	End_Date
B001	B	C00001	2017-06-01 14:00:00	2020-06-02 14:00:00
B002	B	C00002	2015-04-14 18:09:00	2019-03-14 18:09:00
B003	S	C00003	2019-07-08 12:32:00	NULL
B004	P	C00004	2018-03-03 20:56:05	2020-05-06 08:34:22
B005	S	C00005	2016-09-09 21:20:00	NULL
B006	B	C00006	2019-10-10 08:40:00	NULL
B007	S	C00007	2019-10-01 08:44:00	2020-12-30 23:04:50
B008	S	C00008	2017-05-03 13:20:00	2018-04-02 12:32:17
B009	P	C00009	2018-10-03 14:54:00	NULL
B010	S	C00010	2017-05-01 09:09:10	NULL
B011	S	C00011	2018-10-10 08:40:00	NULL
B012	P	C00012	2019-01-10 08:42:00	2020-01-10 09:10:00
B013	P	C00013	2016-10-21 21:02:00	NULL
B014	P	C00014	2017-05-01 13:50:00	2019-07-01 23:45:00
B015	B	C00015	2017-06-09 12:20:00	NULL
B016	B	C00016	2019-04-19 10:20:00	2020-03-19 10:30:00
B017	S	C00017	2018-02-27 21:50:00	NULL
B018	P	C00018	2017-06-13 21:54:00	2020-06-13 20:00:00
B019	B	C00019	2018-11-29 18:35:00	2020-10-20 18:30:00
B020	B	C00020	2019-07-10 08:40:00	2020-08-13 08:10:00

Week 3

Create two Views for different use cases, write five SELECT queries without Joins on the tables. Explain the purpose of each View and query.

1. **Views:**

A. **Purpose of this view is to see the customer details alongwith billing details**
CREATE VIEW CustomerBillingView AS
SELECT c.CustomerID, c.First_name, c.Last_name, c.Email, c.Customer_Status,cb.BillingID, cb.Type_of_Plan, cb.Start_Date, cb.End_Date FROM Customer c INNER JOIN Customer_Billing cb ON c.CustomerID=cb.CustomerID;

SELECT * FROM CustomerBillingView;

CustomerID	First_name	Last_name	Email	Customer_Status	BillingID	Type_of_Plan	Start_Date	End_Date
C00001	Alby	Carter	abcarter@gmail.com	Not Active	B001	B	2017-06-01 14:00:00	2020-06-02 14:00:00
C00002	Damien	Dexter	dexter@gmail.com	Not Active	B002	B	2015-04-14 18:09:00	2019-03-14 18:09:00
C00003	Carl	Christy	ccool@gmail.com	Active	B003	S	2019-07-08 12:32:00	NULL
C00004	David	Warner	warner@gmail.com	Not Active	B004	P	2018-03-03 20:56:05	2020-05-06 08:34:22
C00005	Steven	Hughes	steven01@gmail.com	Active	B005	S	2016-09-09 21:20:00	NULL

B. Purpose of this view is to see the complaints about content quality

```
CREATE VIEW ContentQualityComplaint AS
SELECT Complaint_description, Complaint_date, Complaint_Status FROM Customer
WHERE Complaint_description LIKE '%quality%'
OR Complaint_description like '%HD%'
OR Complaint_description like '%Resolution%';

SELECT * FROM ContentQualityComplaint;
```

Complaint_description	Complaint_date	Complaint_Status
Resolution is not good, Every time I search a m…	2017-07-01 14:00:00,2019-06-01 14:00:00	Open,Open
I cannot access a certain TV Show., The quality…	2018-03-03 20:56:05,2019-03-03 20:56:05	Open, Open
I always had the high def plan but times are tou…	2020-10-03 14:54:00	Closed
Not loads of quality content and the video qualit…	2019-06-13 21:54:00	Closed

2. Write five SELECT queries without Joins on the table :

A. Purpose of this query is to see the countries and the number of complaints for each

```
SELECT Country, COUNT(Complaint_description) AS TotalComplaints
FROM Customer
GROUP BY Country
ORDER BY TotalComplaints DESC ;
```

Country	TotalComplaints
USA	4
Australia	3
Germany	3
UK	2
Canada	1

B. Purpose of this query is to determine the total number of customers in each plan type

```
SELECT Type_of_Plan, COUNT(Type_of_Plan) AS Total_customers FROM
CustomerBillingView
GROUP BY Type_of_Plan
ORDER BY Total_customers DESC;
```

Type_of_Plan	Total_customers
B	7
S	7
P	6

C. Purpose of this query is to evaluate which top 5 customers has viewed the most content

```
SELECT ProfileID, COUNT(ContentID) AS Total_Content_Views
FROM Customer_Viewing_History
GROUP BY ProfileID
ORDER BY Total_Content_Views DESC
LIMIT 5;
```

ProfileID	Total_Content_Views
P00009	3
P00003	2
P00010	2
P00006	2
P00017	1

D. Purpose of this query is to see the details of customer complaints that were due to the content being 'boring'

```
SELECT * FROM Customer
WHERE Complaint_description LIKE '%boring%'
ORDER BY CustomerID;
```

CustomerID	First_Name	Last_Name	Email	Phone	Credit_Card_Name	Credit_Card_Bank	Credit_Card_Number	Customer_Status	PlanID	Country	Complaint_date	Complaint_description
C00017	David	Hernandez	dfernandes@yahoo.com	408236679	David Hernandez	Bank of America	2080151298	Active	S	Germany	2020-02-27 21:50:00	Its getting boring day by day. Not many great sh...

E. Purpose of this query is to determine the total content from each genre

```
SELECT Genre, COUNT(Genre) AS Num_Genre FROM Content
GROUP BY Genre
ORDER BY Num_Genre DESC;
```

Genre	Num_Genre
Drama	4
Documentary	3
Romance	3
Science Fiction	3
Horror	2
Holiday	2
Thriller	2
True Crime	1

Week 4

Creation of Normalised Tables:

For our own analysis purposes, we have normalized the tables we have created as part of our Week 2 assignment. The code below shows the new normalized tables.

1. **Creation of Customer_complaints' table (Normalized)**

```
CREATE TABLE Customer_Complaints (ComplaintID VARCHAR(50),
CustomerID VARCHAR(20),
Complaint_date VARCHAR(50),
Complaint_description VARCHAR(1000),
Complaint_Status VARCHAR(20),
CONSTRAINT PKComplaintID PRIMARY KEY (ComplaintID));
```

Field	Type	Null	Key	Default
ComplaintID	varchar(50)	NO	PRI	NULL
CustomerID	varchar(20)	YES	MUL	NULL
Complaint_date	varchar(50)	YES		NULL
Complaint_description	varchar(1000)	YES		NULL
Complaint_Status	varchar(20)	YES		NULL

Referencing CustomerID as FK in Customer_Complaints:

```
ALTER TABLE Customer_Complaints
ADD CONSTRAINT FKCustomerID
FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
```

An example of the INSERT code is as below:

```
INSERT INTO Customer_Complaints VALUES
('CM001','C00001','2017-07-01 14:00:00','Resolution is not good','Open');
```

ComplaintID	CustomerID	Complaint_date	Complaint_description	Complaint_Status
CM001	C00001	2017-07-01 14:00:00	Resolution is not good	Open
CM002	C00002	2015-04-14 18:09:00	Why are monthly charges increasing?	Open
CM003	C00003	2019-07-08 12:32:00	Netflix Original releases are way too slow.	Closed
CM004	C00004	2018-03-03 20:56:05	I cannot access a certain TV Show.	Open
CM005	C00005	2016-09-09 21:20:00	I am experiencing trouble with accessing a movie.	Closed

2. **Creation of the Cast table (normalized):**
   CREATE TABLE Movie_Cast (CastID VARCHAR(50),
   Director VARCHAR(50) NOT NULL,
   Actor VARCHAR(50),
   ActressVARCHAR(50), CONSTRAINT PKCastID PRIMARY KEY (CastID));

Field	Type	Null	Key	Default
CastID	varchar(50)	NO	PRI	NULL
Director	varchar(50)	NO		NULL
Actor	varchar(50)	YES		NULL
Actress	varchar(50)	YES		NULL

Referencing CastID as FK in Content:
ALTER TABLE Content
ADD CONSTRAINT FKCastID
FOREIGN KEY (CastID) REFERENCES Movie_Cast(CastID);

An example of the INSERT code is as below:
INSERT INTO Movie_Cast VALUES ('C01','Angel Gomez Hernandez','Rodolfo Sancho','Ana Fernandez');

CastID	Director	Actor	Actress
C01	Angel Gomez Hernandez	Rodolfo Sancho	Ana Fernandez
C02	Andrea Santiago	Tom Brady	Mila Kinus
C03	Steven Spielberg	Matt Damon	Cate Blanchett
C04	Christopher Nolan	Clint Eastwood	NULL
C05	Michael Bay	Brad Pitt	Sandra Bullock

Code to change into Normalized tables:

1. **Changing the Customer Table:**
   ALTER TABLE Customer DROP Complaint_date;
   ALTER TABLE Customer DROP Complaint_description;
   ALTER TABLE Customer DROP Complaint_Status;

Field	Type	Null	Key	Default
CustomerID	varchar(20)	NO	PRI	NULL
First_Name	varchar(50)	NO		NULL
Last_Name	varchar(50)	NO		NULL
Email	varchar(30)	NO		NULL
Phone	bigint	NO		NULL
Credit_Card_Name	varchar(30)	NO		NULL
Credit_Card_Bank	varchar(20)	NO		NULL
Credit_Card_Number	bigint	NO		NULL
Customer_Status	varchar(10)	YES		NULL
PlanID	varchar(5)	YES		NULL
Country	varchar(20)	YES		NULL

2. **Changing the Content Table:**
   ALTER TABLE Content DROP Director;
   ALTER TABLE Content DROP Actor;
   ALTER TABLE Content DROP Actress;

Field	Type	Null	Key	Default
ContentID	varchar(50)	NO	PRI	NULL
Title	varchar(50)	NO		NULL
CastID	varchar(50)	YES	MUL	NULL
Netflix_Original	char(10)	NO		NULL
Genre	varchar(20)	NO		NULL
Video_Type	varchar(50)	NO		NULL
Release_Date	datetime	NO		NULL
Duration	smallint	YES		NULL
No_of_Seasons	smallint	YES		NULL
No_of_Episodes	smallint	YES		NULL
Production_Co…	varchar(50)	NO		NULL
Licensing_Cost	bigint	YES		NULL
Licensing_Years	smallint	YES		NULL

1. **Write five SQL statements using Joins, SubQuery concepts to answer your business questions. Explain how your statements answer your questions.**

A. **How can we retain customers? Why are they choosing Netflix?**

i. **Finding out what users are watching the most of**

```
SELECT cvh.ContentID, ct.Title , ct.Genre, ct.Netflix_Original, ct.Video_Type,
SUM(ct.Duration) AS Total_Duration,
COUNT(DISTINCT(cp.ProfileID)) AS Total_Profile_Views,
SUM(p.Rate) AS Revenue_From_Customers_Watching
FROM Customer_Viewing_History cvh
INNER JOIN Content ct ON cvh.ContentID = ct.ContentID
LEFT JOIN Customer_Profiles cp ON cvh.ProfileID = cp.ProfileID
LEFT JOIN Customer_Billing cb ON cp.CustomerID = cb.CustomerID
LEFT JOIN Plans p ON cb.Type_of_Plan = p.PlanID
GROUP BY ContentID
ORDER BY Total_Duration DESC
LIMIT 5;
```

ContentID	Title	Genre	Netflix_Original	Video_Type	Total_Duration	Total_Profile_Views	Revenue_From_Customers_Watching
00000M8	Luck in the night	Romance	No	Movie	210	2	22.98
00000M10	Strangers of hope	Drama	No	Movie	145	1	13.99
00000T3	what happened to monday	Science Fiction	Yes	Movie	128	1	13.99
00000M2	Disclosure	Documentary	No	Movie	120	1	17.99
00000M5	Moonlight	Drama	No	Movie	120	1	13.99

ii. **Viewing customer membership duration and time they spent on Netflix**

```
CREATE VIEW Detailed_Customer_View AS
SELECT c.CustomerID, SUM(cvh.Duration) AS Total_Viewing_Time
FROM Customer c
INNER JOIN Customer_Profiles cp ON c.CustomerID = cp.CustomerID
INNER JOIN Customer_Viewing_History cvh ON cp.ProfileID = cvh.ProfileID
GROUP BY CustomerID;
```

CustomerID	Total_Viewing_Time
C00001	338
C00002	718
C00003	105
C00004	245
C00006	35
C00005	1184
C00010	607
C00007	100
C00008	45

```
SELECT cb.CustomerID, DATEDIFF(End_Date, Start_Date)/365 AS
Years_of_Membership, Total_Viewing_Time
FROM Customer_Billing cb
LEFT JOIN Detailed_Customer_View dtv ON cb.CustomerID = dtv.CustomerID
ORDER BY Years_of_Membership DESC;
```

CustomerID	Years_of_Membership	Total_Viewing_Time
C00002	3.9178	718
C00001	3.0055	338
C00018	3.0027	NULL
C00004	2.1781	245
C00014	2.1671	NULL
C00019	1.8932	NULL
C00007	1.2493	100
C00020	1.0959	NULL
C00012	1.0000	NULL
C00016	0.9178	NULL
C00008	0.9151	45
C00003	NULL	105
C00005	NULL	1184
C00006	NULL	35
C00009	NULL	NULL
C00010	NULL	607
C00011	NULL	NULL
C00013	NULL	NULL
C00015	NULL	NULL
C00017	NULL	NULL

Recommendation:

Netflix Originals are not necessarily what is driving customers to Netflix as the most viewed movies are not Netflix Originals. We can also see that some customers who have cancelled their memberships (non-null values in years of membership) have spent no time viewing anything on Netflix. This could be that

they were dissatisfied with the options of movies/TV shows on Netflix. Perhaps, Netflix could offer a more popular variety of materials on their website to attract and retain customers.

B. Are customers willing to pay higher prices for better resolution?

Purpose of this view is to get the complaints about streaming resolution?

```
SELECT Complaint_description, Complaint_date, Complaint_Status FROM
Customer_Complaints
WHERE Complaint_description LIKE '%quality%'
OR Complaint_description like '%HD%'
OR Complaint_description like '%Resolution%';
```

Complaint_description	Complaint_date	Complaint_Status
Resolution is not good	2017-07-01 14:00:00	Open
I always had the high def plan but times are tou…	2020-10-03 14:54:00	Closed
The quality of these movies are not HD more lik…	2019-03-03 20:56:05	Open
Not loads of quality content, and the video qualit…	2019-06-13 21:54:00	Closed

Viewing customers complained about streaming quality but have not upgraded to higher plans and left Netflix.

```
SELECT cmp.CustomerID, cmp.Complaint_description, c.PlanID,
c.Customer_Status FROM Customer_Complaints cmp
JOIN Customer c ON cmp.CustomerID = c.CustomerID
WHERE (Complaint_description LIKE '%quality%'
        OR Complaint_description like '%HD%'
        OR Complaint_description like '%streaming%'
        OR Complaint_description like '%Resolution%') AND PlanID = 'B';
```

CustomerID	Complaint_description	PlanID	Customer_Status
C00001	Resolution is not good	B	Not Active
C00018	Not loads of quality content, and the video qualit…	B	Not Active

Recommendation:

Customers are not willing to pay more to get better streaming resolution. Netflix could have their customers use a better resolution plan but cheaper price.

C. How many customers have left Netflix after raising complaints?

i. Purpose of this query is to get the number of not active customers who have open complaint status

```
SELECT COUNT(*) FROM Customer WHERE Customer_Status='Not Active'
AND CustomerID IN (SELECT CustomerID FROM Customer_Complaints
WHERE Complaint_Status='Open');
```

Count_customers
8

ii. Purpose of this query is to prove that customers left Netflix after having unresolved complaints for long durations especially loyal customers.

```
SELECT cmp.CustomerID, cmp.Complaint_Status, cmp.Complaint_date,
cb.End_Date, c.Customer_Status,
TIMESTAMPDIFF(MONTH, cmp.Complaint_date, cb.End_Date) AS Lead_time
FROM Customer_Complaints cmp
JOIN Customer c ON cmp.CustomerID=c.CustomerID
JOIN Customer_Billing cb ON cb.CustomerID=c.CustomerID
WHERE c.Customer_Status='Not Active' AND cmp.Complaint_Status='Open';
```

CustomerID	Complaint_Status	Complaint_date	End_Date	Customer_Status	Lead_time
C00001	Open	2017-07-01 14:00:00	2020-06-02 14:00:00	Not Active	35
C00001	Open	2019-06-01 14:00:00	2020-06-02 14:00:00	Not Active	12
C00002	Open	2015-04-14 18:09:00	2019-03-14 18:09:00	Not Active	47
C00002	Open	2017-04-14 18:09:00	2019-03-14 18:09:00	Not Active	23
C00004	Open	2018-03-03 20:56:05	2020-05-06 08:34:22	Not Active	26
C00004	Open	2019-03-03 20:56:05	2020-05-06 08:34:22	Not Active	14
C00007	Open	2020-10-01 08:44:00	2020-12-30 23:04:50	Not Active	2
C00008	Open	2017-08-05 13:20:00	2018-04-02 12:32:17	Not Active	7
C00008	Open	2018-03-03 13:20:00	2018-04-02 12:32:17	Not Active	0
C00016	Open	2020-01-19 10:20:00	2020-03-19 10:30:00	Not Active	2
C00019	Open	2020-09-29 12:35:00	2020-10-20 18:30:00	Not Active	0
C00020	Open	2020-06-10 08:50:00	2020-08-13 08:10:00	Not Active	2

Recommendation:

By resolving the complaints in about 48hrs duration, Netflix can retain their customers which can impact their profit.

2. **Write one trigger and one stored procedure.**

Stored procedure:

```
DELIMITER  $$
CREATE PROCEDURE GetCustomers()
BEGIN
  SELECT CustomerID, First_name,Last_name, Email,Phone
  FROM Customer;
END $$
DELIMITER ;
CALL GetCustomers();
```

Trigger:

```
# Create a new table - PlanCountInsert
DROP TABLE  IF EXISTS PlanCountInsert;
CREATE TABLE PlanCountInsert (
PlanID CHAR(20),
Customer_count INT);
```

PlanCountInsert Table Before the Insert on Customer Billing:

PlanID	Customer_count
S	7
P	6
B	7

```
# Create a trigger
DELIMITER $$
DROP TRIGGER IF EXISTS  PlanCountInsertTrigger$$
CREATE TRIGGER PlanCountInsertTrigger
AFTER INSERT ON Customer_Billing FOR EACH ROW
IF EXISTS (SELECT * FROM PlanCountInsert WHERE PlanID=New.Type_of_Plan)
THEN UPDATE PlanCountInsert SET Customer_count = Customer_count + 1 WHERE
PlanID=New.Type_of_Plan;
END IF $$
DELIMITER ;

# Insert new rows into Customer_Billing
INSERT INTO Customer_Billing VALUES ('B0022','B','C000022','2017-06-01
14:00:00','2020-06-02 14:00:00');

SELECT * FROM PlanCountInsert;
```

PlanID	Customer_count
B	8
P	6
S	7