جامعة شقراء
Shaqra University

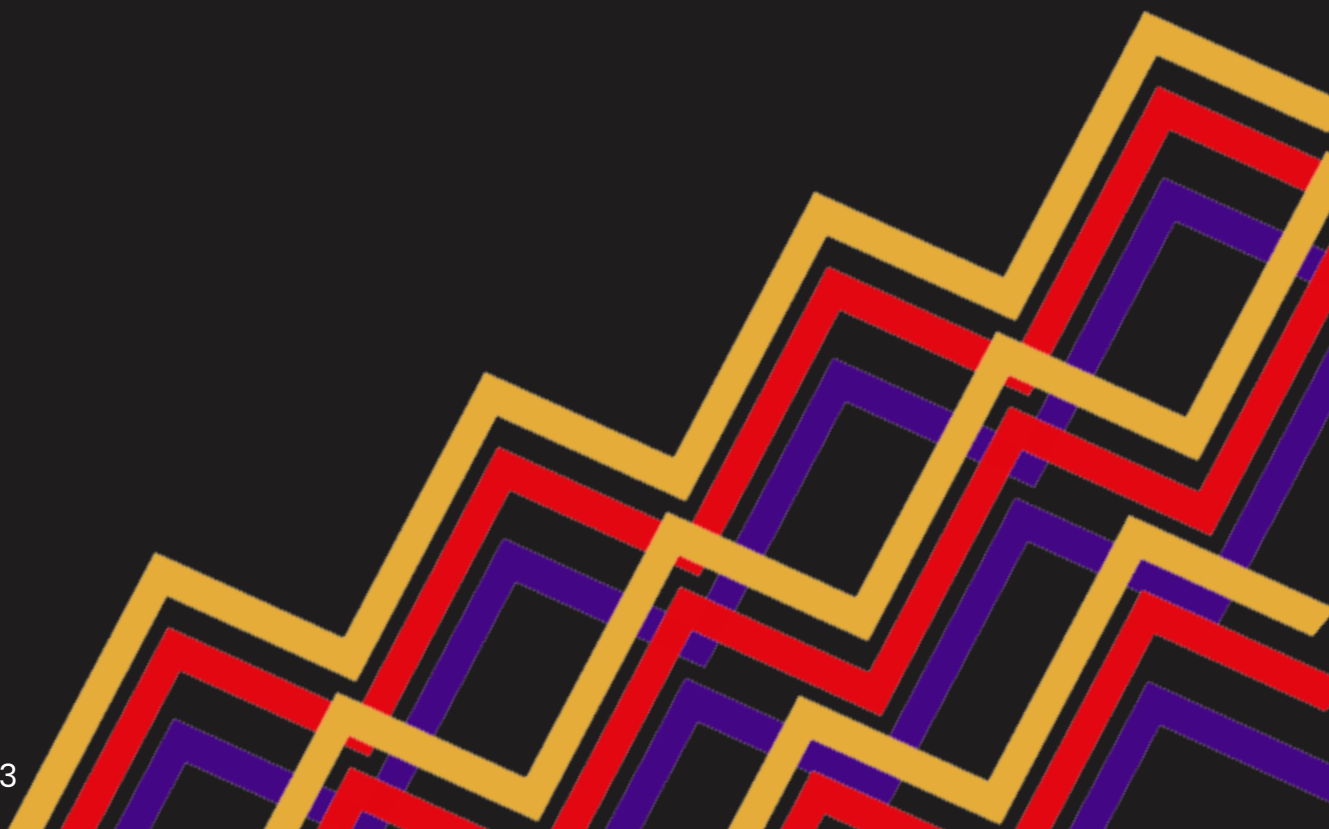
DataBase Management System of Netflix

Prepared by

Lina Alfawzan
Rawabi Albaqami

Proposed to

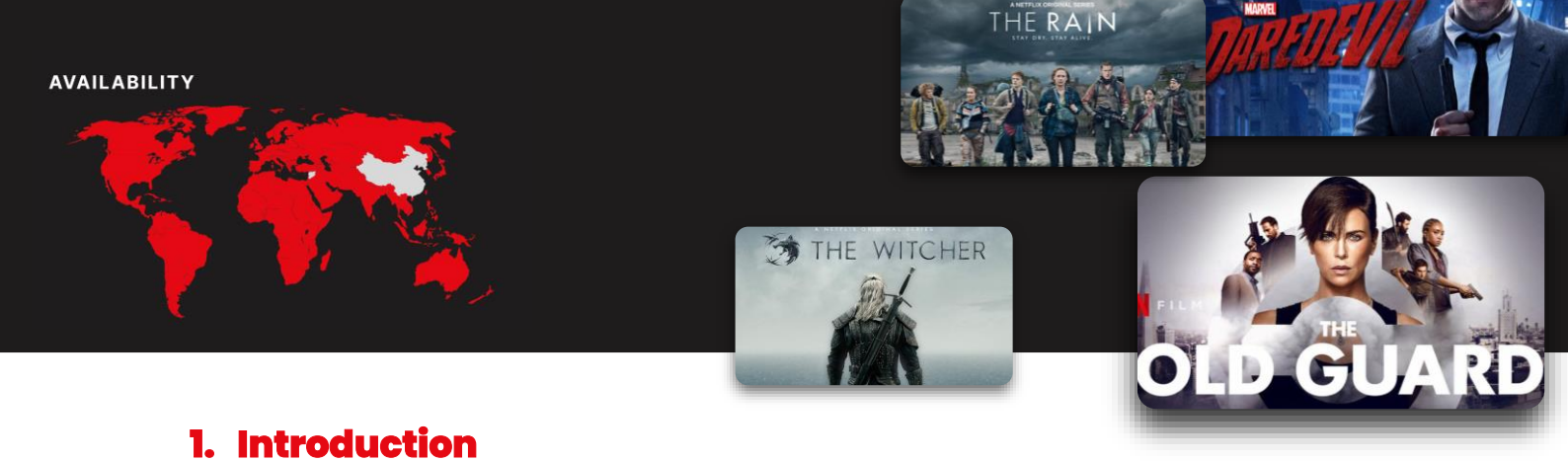
Dr. Mohamed Elfaki



Year 1444 - 2023

Table of content

1. Introduction	**3**
1.1 Problem	3
1.2 Objectives	3
1.3 Intended Audience	3
2. Requirements Elicitation	**4**
2.1 Design and Implementation Constraints	5
2.2 Scenario: Watching a movie	5
3. System Design	**6**
3.1 Enhanced Entity Relationship (EER)	6
3.2 Database schema	7
4. Database System Architecture	**10**
4.1 System architecture type	10
4.2 System network type	11
5. Implementation	**13**
5.1 ER – Diagram	13
5.2 MySQL queries	13
6. Conclusion	**15**
6.1 Acknowledgement	15
References	**16**



1. Introduction

People love TV content, but they don't love the linear TV experience, where channels present programs only at particular times on non-portable screens with complicated remote controls. Now internet TV - which is on-demand, personalized, and available on any screen -is replacing linear TV. Changes of this magnitude are rare. Radio was the dominant home entertainment media for nearly 50 years until linear TV took over in the 1950's and 1960's.Linear video in the home was a huge advance over radio, and very large firms emerged to meet consumer desires over the last 60 years. The new era of internet TV, which began a decade ago, is likely to be very big and enduring also, given the flexibility and ubiquity of the internet around the world.

NETFLIX is the world's leading internet TV network, with more than 100 million members worldwide enjoying 125 million hours of TV shows and movies each day, including original series, documentaries, and feature films.

1.1 Problem

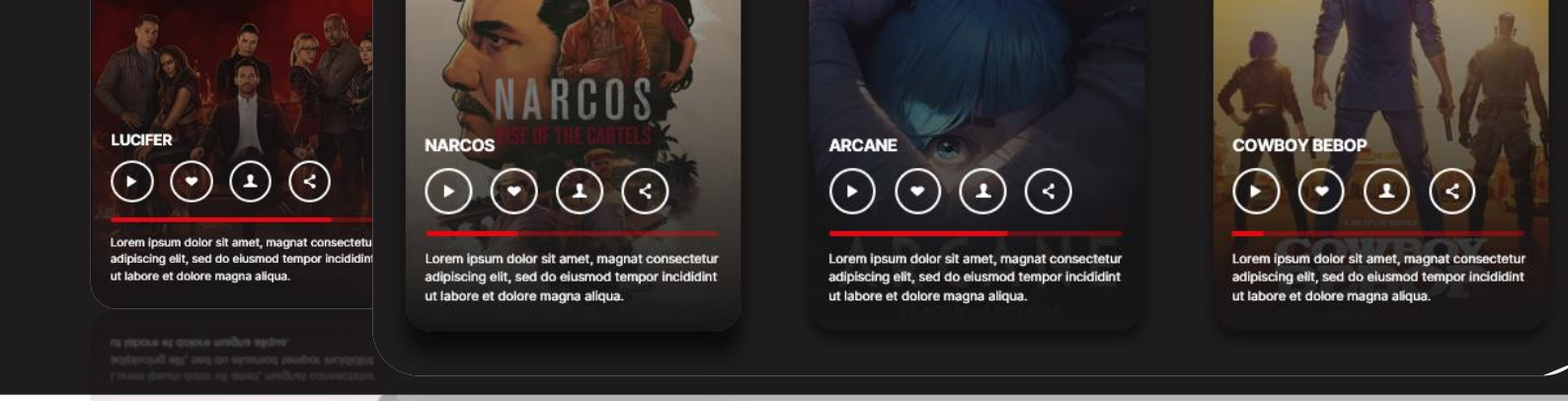
The problem with traditional TV is that you cannot control when and where your favourite show is shown. Also, there are many existing platforms to watch series and TV shows, but there is no uninterrupted streaming platform that is free of ads and viruses even in the lowest-priced basic plan.

1.2 Objectives

To develop a platform that gives our users control over what they want to watch whatever their taste, no matter where they live, we give you access to the best TV series, documentaries and feature films, in one simple subscription.

1.3 Intended Audience

NETFLIX's target market is young, tech-savvy users and anyone with digital connectivity. The audience of Netflix is from diverse age groups and demographics. However, most of the audience are teenagers, college-goers, entrepreneurs, working professionals, etc.



2. Requirements Elicitation

The following table represents the categorization of requirements captured for **NETFLIX** system:

SNO	Requirements	Type	Priority
R1	Users should be able to share videos.	Functional requirements	Must have
R2	The content team should be able to upload new videos (movies, tv shows episodes, and other content).	Functional requirements	Must have
R3	Users should be able to search for videos using titles or tags.	Functional requirements	Must have
R4	High availability with minimal latency.	Nonfunctional requirements	Should have
R5	High reliability, no uploads should be lost.	Nonfunctional requirements	Must have
R6	The system should be scalable and efficient.	Nonfunctional requirements	Must have
R7	Performance.	Nonfunctional requirements	Must have
R8	Safety.	Nonfunctional requirements	Must have
R9	Security.	Nonfunctional requirements	Must have
R10	Software Quality.	Nonfunctional requirements	Must have
R11	Record metrics and analytics of videos.	Extended requirements	Should have
R12	Certain content should be geo-blocked.	Extended requirements	Must have
R13	Resume video playback from the point user left off.	Extended requirements	Should have

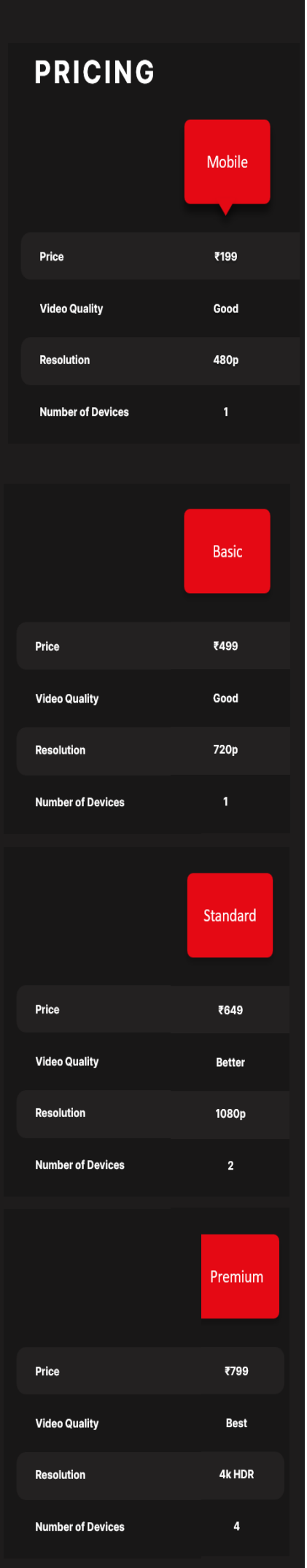


Figure (1)
Pricing plans for Netflix

2.1 Design and Implementation Constraints

Server capacity is how many users can access or can be online once. More is the number of users more will be the network traffic and hence the serve comes in a down state. Personal firewall and updating is a tough task, it should be such that it should not block the network traffic, making the system slower. Firewall of the server should not collide with the firewall of the user system.

2.2 Scenario: Watching a movie

User Action	Providing movie Management
A customer logs in to an online portal to watch a movie.	Not applicable.
The customer does not have an account.	The customer creates an account by providing the personal details and becomes the member. On creating an account, the customer record is created and maintained in Sterling Order Management.
The customer is guided to the payment screen where the customer can select a preferred plan and payment method to pay for the service.	The selected payment method and plan is added to the draft order.
The customer confirms.	An email notification is sent to the customer and the account is successfully generated.
The customer performs a movie search by entering the appropriate search criteria.	As a result of the search, the movie details are displayed.

3. System Design

Effective database design means that your software is capable of managing and consolidating all the data generated and relied upon by your business. A good database design will allow your organization to develop a clear structure for the way in which data is stored and managed by every person or application using it.

3.1 Enhanced Entity Relationship (EER)

An Enhanced Entity Relationship (EER) Diagram for **NETFLIX** is a graphical representation of the database and how it relates to different entities. It is used as a visual representation of the relationships between entities and how they interact with various data elements. EER Diagrams makes it easier to track information, find errors and design efficient databases. The diagram includes entities like Customer, Subscription, Payment, Content, and Stars.

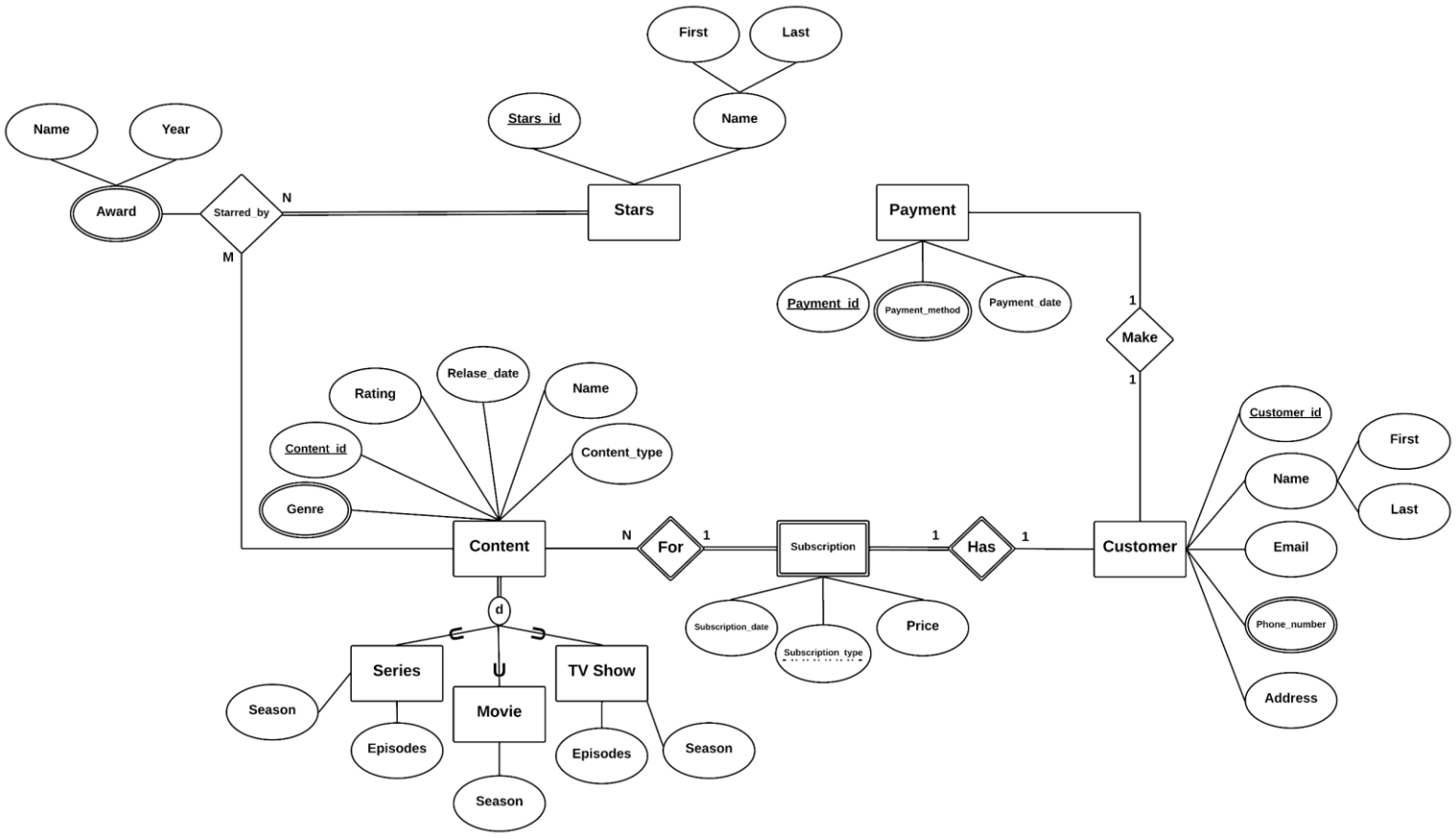


Figure (2)

(EER) Diagram for **NETFLIX**

3.2 EER-to-Relational Mapping Algorithm

Customer

Customer_id	First_name	Last_name	Email	Address	Subscription_type

Phone_number

Phone_number	Customer_id

Make

Customer_id	Payment_id

Payment

Payment_id	Payment_date

Payment_method

Payment_method	Payment_id

Subscription

Customer_id	Content_id	Subscription_type	Subscription_date	Price

Content

Content_id	Rating	Release_date	Name	Content_type	Subscription_type	S_season	M_season	TV_season	S_episodes	TV_episodes

Genre

Content_id	Genre

Starred_by

Stars_id	Content_id	Award_name	Award_year

Stars

Stars_id	First_name	Last_name

Figure (3)

Relational Mapping for **NETFLIX**

Step 1: Mapping of Regular Entity Types.

We create the relations *Customer, Payment, Content* and *Stars* in the relational schema corresponding to the regular entities in the ER diagram. *Customer_id, Payment_id, Content_id, and Stars_id* are the primary keys for the relations *Customer, Payment, Content and Stars* as shown

Step 2: Mapping of Weak Entity Types.

Create the relation *Subscription* in this step to correspond to the weak entity type *Subscription*. Include the primary keys *Customer_id* and *Content_id* of the *Customer* and *Content* relations as a foreign key attribute of *Subscription*. The primary key of the *Subscription* relation is the combination {*Customer_id, Content_id, Subscription_type*} because *Subscription_type* is the partial key of *Subscription relation*.

Step 3: Mapping of Binary 1:1 Relation Types.

In the EER there are two 1:1 relationship, the first one *Has relationship* we use:
Foreign Key approach: we choose an entity type with total participation which is *Subscription* relation, the partial key of *Subscription* named *Subscription_type* included in *Customer relation* as foreign key.

The second one *Make* relationship we use:
Cross-reference or relationship relation option: set up a third relation *Make* for the purpose of cross-referencing the primary keys of the two relations *Customer* and *Payment* named *Customer_id* and *Payment_id.*

Step 4: Mapping of Binary 1:N Relationship Types.

1:N relationship *For*. For *For* relationship, we include the partial key *Subscription_type* of the *Subscription relation* as foreign key in the *Content relation.*

Step 5: Mapping of Binary M:N Relationship Types.

The M:N relationship type *Starred_by* from the ER diagram is mapped by creating a relation *Starred_by* in the relational database schema. The primary keys of the *Content* and *Stars* relations are included as foreign keys in *Starred_by* and named *Content_id* and *Star_id*, respectively.

Step 6: Mapping of Multivalued attributes.

The relations *Phone_number, Payment_method, and Genre* are created. We did not create an *award* relation to avoid redundancy because it is already existing in the database in *starred_by* relation.

Phone_number:
The attribute *Phone_number* represents the multivalued attribute, while *Customer_id* as foreign key-represents the primary key of the *Customer* relation. The primary key of *Phone_number* is the combination of *{Customer_id, Phone_number}.*

Payment_method:
The attribute *Payment_method* represents the multivalued attribute, while *Payment_id* as foreign key-represents the primary key of the *Payment* relation. The primary key of *Payment_method* is the combination of {*Payment_id, Payment_method*}.

Genre:
The attribute *Genre* represents the multivalued attribute, while *Content_id* as foreign key-represents the primary key of the *Content* relation. The primary key of *Genre* is the combination of {*Content_id, Genre*}.

Step7: Options for Mapping Specialization or Generalization.

Convert specialization with 3 subclasses {*Series*, *TV show*, *Movie*} and generalized superclass *Content*. **Using option C: single relation with one type attribute.**

Create a single relation *Content*. The attribute *Content_type* is called a type (or discriminating) attribute that indicates the subclass to which each tuple belongs.

4. Database system architecture

A Database Architecture is a representation of DBMS design. It helps to design, develop, implement, and maintain the database management system. A DBMS architecture allows dividing the database system into individual components that can be independently modified, changed, replaced, and altered. It also helps to understand the components of a database.

A Database stores critical information and helps access data quickly and securely. Therefore, selecting the correct Architecture of DBMS helps in easy and efficient data management.

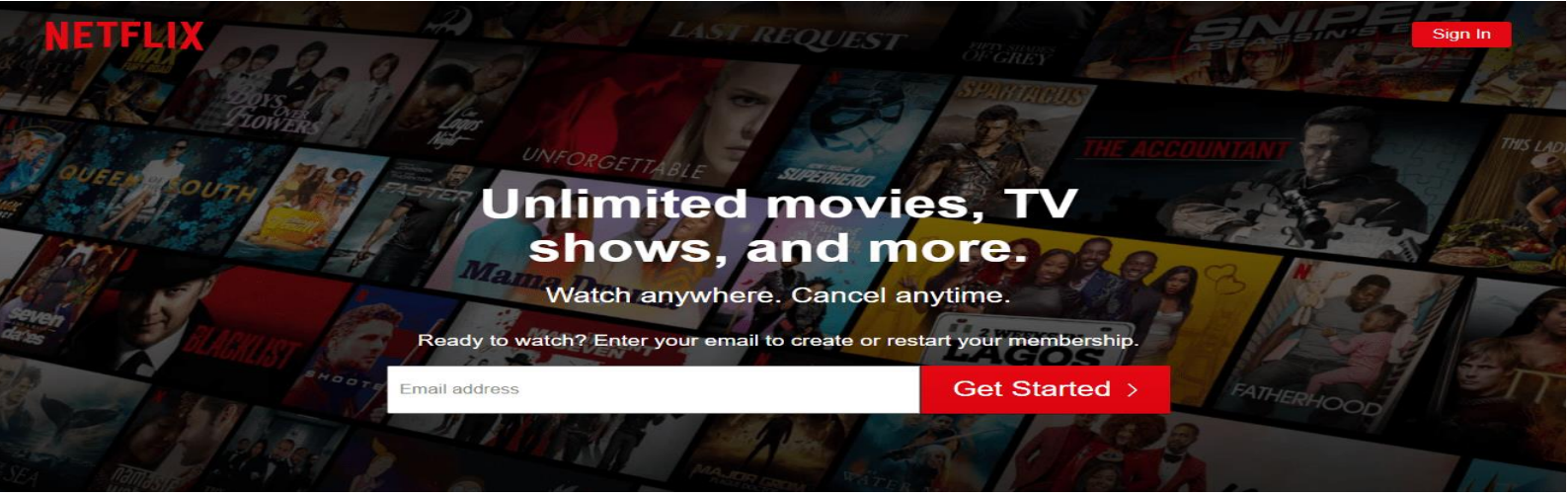
4.1 System architecture type

Today business operations have become decentralized, to a global level. The growth of the internet has become one of the primary needs for instantaneous data access. We can see an example of this today with the widespread use of smartphones and mobile devices.

High demand smartphone apps require fast, wireless access to databases from multiple locations. **Distributed Database Management Systems** (DDBMS) have evolved to meet this demand, and this is one of the primary advantages of using a DDBMS in **NETFLIX**.

NETFLIX uses a DDBMS so that data can be stored locally in locations with the highest demand. This improves access time. Additionally, when accessing data which is what happens when you select a movie on Netflix, the query can be processed in multiple database locations, taking advantage of the parallel CPU processing power available in a DDBMS. This reduces hardware demands and improves efficiency.

Whereas in a **centralized database**, data access is limited to the hardware capabilities of the particular system. These single centralized systems often exhibit latency when system demands are high, so it is not even a viable option for **NETFLIX**.



4.2 System network type

The network type used in designing **NETFLIX** is a **mesh network** which is a network in which devices or nodes are linked together, branching off other devices or nodes. These networks are set up to efficiently route data between devices and clients. They help organizations provide a consistent connection throughout a physical space.

Mesh networks include the following benefits:

- **Increased stability.** Single points of failure don't harm the whole network.
- **Increased range.** Mesh networks can transmit signals over a greater distance. They have fewer dead spots where Wi-Fi signals don't reach.
- **Direct communication.** Nodes can message each other directly. There is no need for intervention from a central access point.
- **Less power is needed for each node.** Each device in the network doesn't need to put out a signal strong enough to reach a central access point.
- **Better security.** If attacked, single nodes are easily replaced.
- **Simpler topology.** Mesh networks require less infrastructure than other types of network's configurations.

5. Implementation

The database management system (DBMS) of **NETFLIX** is done by MySQL database due to effective ability to efficiently store, retrieve and share extremely large measurement data MySQL is a relational DBMS with open sources, freely usable under the GNU General Public License.

5.1 ER – Diagram

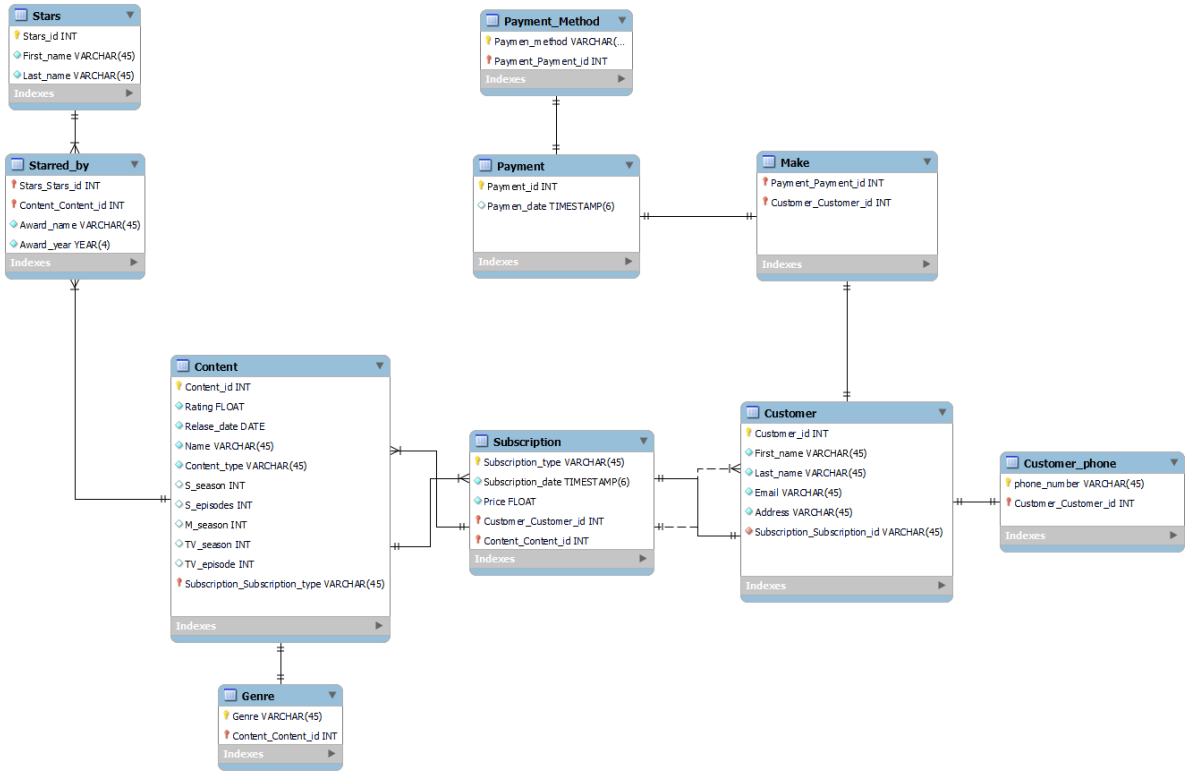


Figure (4)

Entity Relationship Diagram for **NETFLIX**

5.2 MySQL queries

5.2.1 On delete/update cascade

On delete cascade clause in MySQL is used to automatically remove the matching records from the child table when we delete the rows from the parent table.

On update cascade clause in MySQL is used to update the matching records from the child table automatically when we update the rows in the parent table.

We did it on *Stars* and *Starred_by* tables as parent and child.

Step1: Include both update and delete cascade on child table(*Starred_by*):

```
CREATE TABLE IF NOT EXISTS `Netflix`.`Starred_by` (
  `Stars_Stars_id` INT NOT NULL,
  `Content_Content_id` INT NOT NULL,
  `Award_name` VARCHAR(45) NOT NULL,
  `Award_year` YEAR(4) NOT NULL,
  PRIMARY KEY (`Stars_Stars_id`, `Content_Content_id`),
  INDEX `fk_Stars_has_Content_Content1_idx` (`Content_Content_id` ASC)
VISIBLE,
  INDEX `fk_Stars_has_Content_Stars1_idx` (`Stars_Stars_id` ASC) VISIBLE,
  CONSTRAINT `fk_Stars_has_Content_Stars1`
    FOREIGN KEY (`Stars_Stars_id`)
    REFERENCES `Netflix`.`Stars` (`Stars_id`)
    ON DELETE cascade
    ON UPDATE cascade,
  CONSTRAINT `fk_Stars_has_Content_Content1`
    FOREIGN KEY (`Content_Content_id`)
    REFERENCES `Netflix`.`Content` (`Content_id`)
    ON DELETE No Action
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Step2: When we delete or update on parent table(*Stars*) it will affect child table(*Starred_by*) too:

- Tables before:

Stars_Stars_id	Content_Content_id	Award_name	Award_year
1	3	Palme d'Or	2022
1	9	Golden Globe Award for Best Actor	1997
2	5	Asia Best Couple	2016
5	6	Teen Choice Award for Choice TV Actress	2010
7	4	Baeksang Arts Award for Most Popular Actor	2023
9	8	MTV Movie & TV Award for Best Fight	2006
22	10	MTV Movie Award for Best Breakthrough	1999
NULL	NULL	NULL	NULL

starred_by 1

Stars_id	First_name	Last_name
4	Leonardo	DiCaprio
5	Leighton	Meester
6	Penn	Badgley
7	Jinyoung	Park
8	Brad	Pitt
9	Angelina	Jolie
22	Katie	Holmes
NULL	NULL	NULL

stars 1

- Update and delete implemented:

```
delete from stars where stars_id=22;
update stars set stars_id=45 where stars_id=9;
```

- The result:

Stars_Stars_id	Content_Content_id	Award_name	Award_year
1	3	Palme d'Or	2022
1	9	Golden Globe Award for Best Actor	1997
2	5	Asia Best Couple	2016
5	6	Teen Choice Award for Choice TV Actress	2010
7	4	Baeksang Arts Award for Most Popular Actor	2023
45	8	MTV Movie & TV Award for Best Fight	2006
NULL	NULL	NULL	NULL

starred_by 1

Stars_id	First_name	Last_name
4	Leonardo	DiCaprio
5	Leighton	Meester
6	Penn	Badgley
7	Jinyoung	Park
8	Brad	Pitt
45	Angelina	Jolie
NULL	NULL	NULL

stars 1

5.2.2 Trigger

A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server.

We create trigger which convert *subscription_type* into upper case:

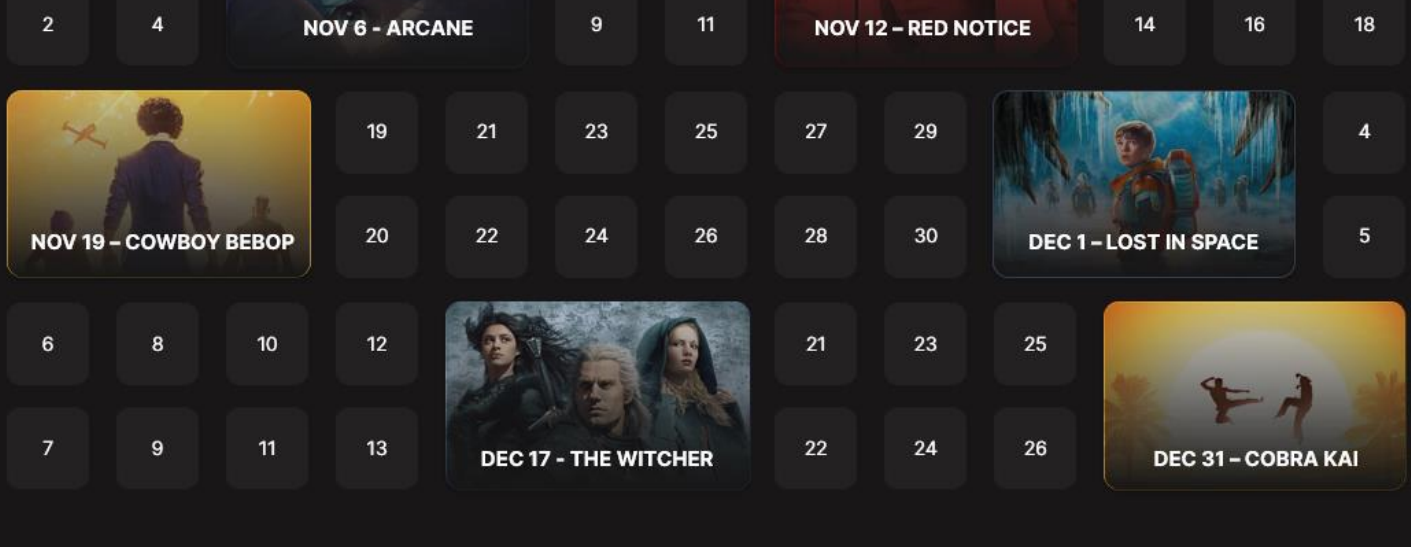
```
create trigger pachage_name
before insert
ON
subscription
for each row
set new.Subscription_type= upper(Subscription_type);
```

5.2.3 Index

Indexes are used to retrieve data from the database more quickly than otherwise. The users cannot see the indexes, they are just used to speed up searches/queries.

We create ascending indexes in most of tables for example in *customer* table:

```
CREATE TABLE IF NOT EXISTS `Netflix`.`Customer` (
  `Customer_id` INT NOT NULL AUTO_INCREMENT,
  `First_name` VARCHAR(45) NOT NULL,
  `Last_name` VARCHAR(45) NOT NULL,
  `Email` VARCHAR(45) NOT NULL,
  `Address` VARCHAR(45) NOT NULL,
  `Subscription_Subscription_id` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`Customer_id`),
  INDEX `Subscription_Subscription_id_idx`
(`Subscription_Subscription_id` ASC) VISIBLE,
  CONSTRAINT `Subscription_Subscription_id`
    FOREIGN KEY (`Subscription_Subscription_id`)
    REFERENCES `Netflix`.`Subscription` (`Subscription_type`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```



6. Conclusion

This project implements a streaming service that offers a wide variety of award-winning TV shows, movies, anime, documentaries, and more on thousands of internet-connected devices. You can watch as much as you want, whenever you want – all for one low monthly price. It can be said with full assurance that if the system is fully implemented, all the advantages of this platform will be desired by all segments of society!

6.1 Acknowledgement

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our project manager, Dr. Mohammed, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project.

References

- ✓ *Diagram maker for developers*. (n.d.). https://www.gleek.io/blog/erd-database-design
- ✓ GeeksforGeeks. (2023, January 12). *System Design Netflix a Complete Architecture*. https://www.geeksforgeeks.org/system-design-netflix-a-complete-architecture/
- ✓ L, N. (2018d, September 8). *NETFLIX system design - Narendra L - Medium*. Medium. https://medium.com/@narengowda/netflix-system-design-dbec30fede8d
- ✓ Walz, E. (n.d.). *How Netflix Uses a Distributed Database Management System to Deliver Your Movies*. https://www.linkedin.com/pulse/how-netflix-uses-distributed-database-management-system-eric-walz
- ✓ BasuMallick, C. (2022, August 12). *Mesh Network Working, Types, Applications - Spiceworks*. Spiceworks. https://www.spiceworks.com/tech/networking/articles/what-is-mesh-network/
- ✓ Blog, N. T. (2022, August 1). Data Mesh — A Data Movement and Processing Platform @ Netflix. *Medium*. https://netflixtechblog.com/data-mesh-a-data-movement-and-processing-platform-netflix-1288bcab2873