Bases de Dados 2021/22

Relatório Projeto Twitch.tv

Grupo P6G9

João Morais (103730)

Ricardo Pinto (103078)

1. Introdução

Para o nosso projeto final da cadeira de Base de Dados, decidimos criar uma base de dados de gestão da Twitch.tv.

A Twitch.tv é uma plataforma de livestreaming onde qualquer pessoa pode criar o seu canal e fazer transmissões em direto de várias formas de entretenimento.

2. Análise de Requisitos

Canal – É identificado por um ID. Tem um nome, data de criação, número de seguidores, número de subscritores, total de dias ativos (em que transmitiu), total de horas transmitidas, total de horas vistas, média de visualizadores e número máximo de visualizadores. Pode seguir e subscrever outros canais, e ser seguido e ser subscrito por outros canais. Pode estar ativo em várias categorias.

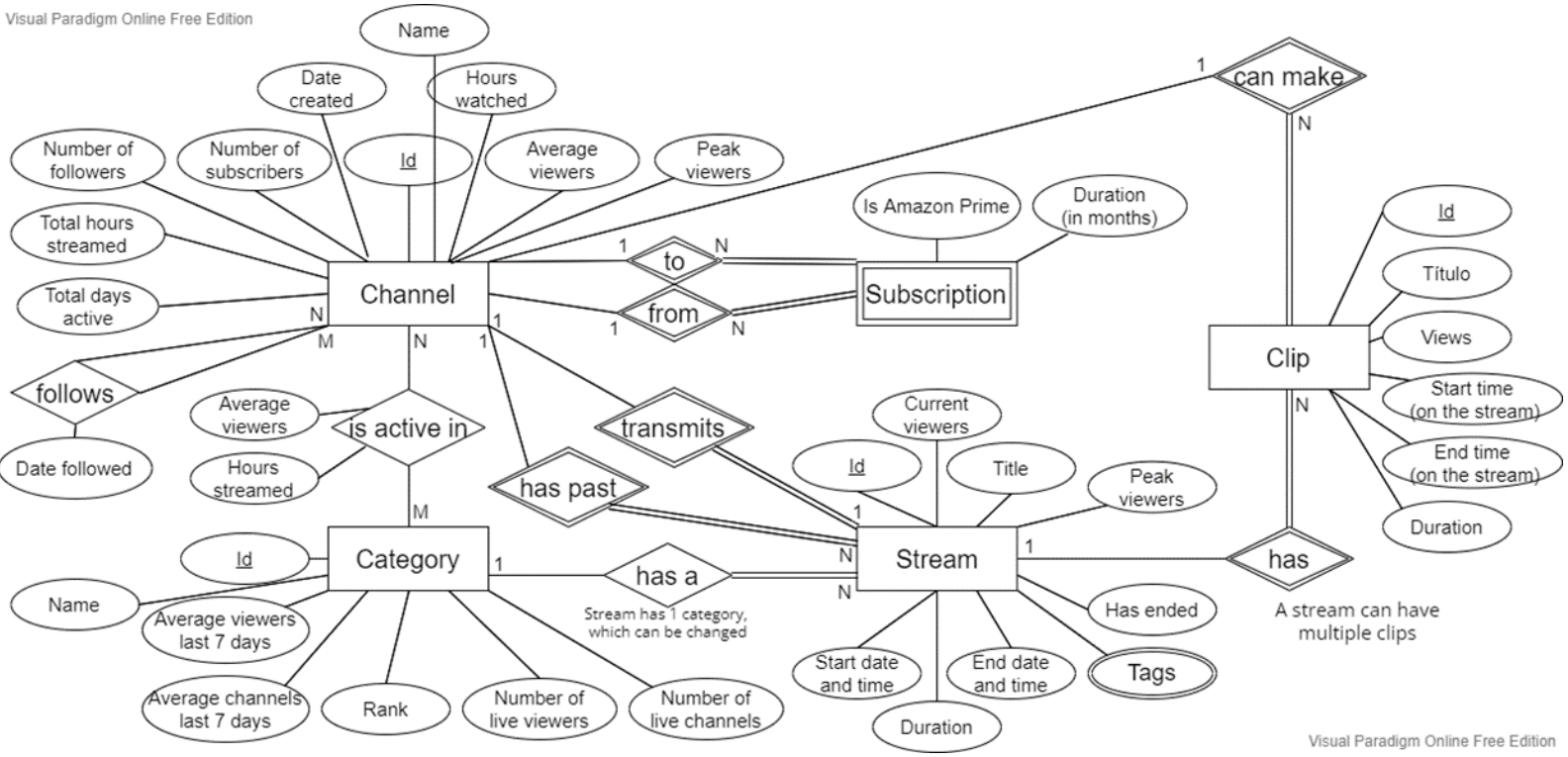
Subscrição – É identificada pelo canal que ofereceu a subscrição e pelo canal ao qual for oferecida a subscrição. Tem a duração em meses e a possibilidade de ser Amazon Prime.

Categoria – É identificada por um ID. Tem um nome, número de visualizadores no momento, número de canais a transmitir no momento, média de visualizadores nos últimos 7 dias, média de canais a transmitir nos últimos 7 dias e um rank.

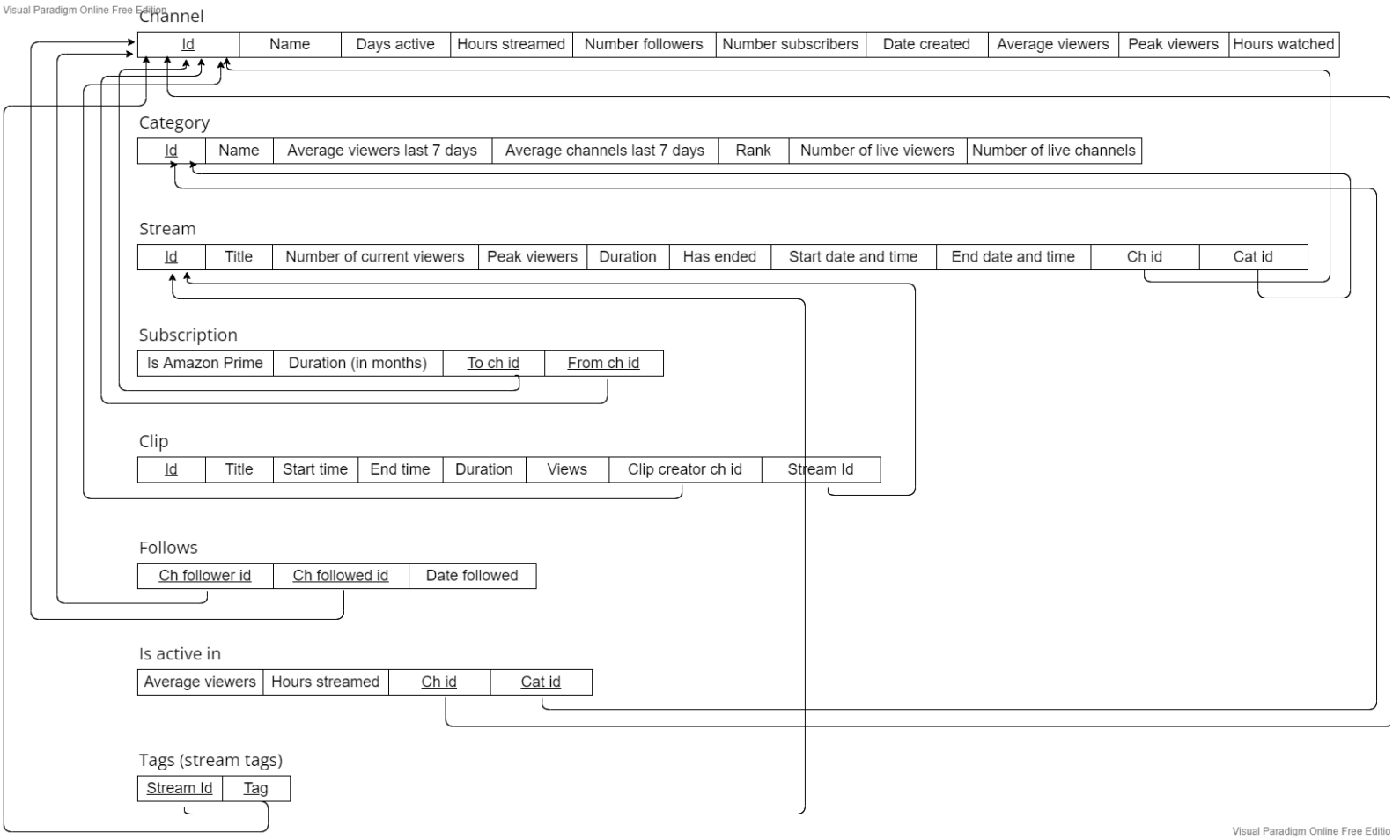
Stream – Um canal pode fazer uma transmissão em direto. É identificada por um ID. Tem um título, número de visualizadores, número máximo de visualizadores, a data e hora de início, a duração em segundos, tags e a possibilidade de já ter acabado. Tem de ter sempre uma categoria que pode ser alterada a qualquer momento.

Clip – É uma pequena parte de uma stream. É identificado por um ID. Tem um título, hora de início, duração em segundos e número de visualizações. Os canais/streams podem ter vários clips. Qualquer canal pode fazer clips de qualquer stream.

3. DER


Visual Paradigm Online Free Edition
Name
Date created
Hours watched
Number of followers
Number of subscribers
Id
Average viewers
Peak viewers
Total hours streamed
Total days active
Channel
follows
Date followed
Average viewers
Hours streamed
is active in
transmits
has past
to
from
Subscription
Is Amazon Prime
Duration (in months)
can make
Clip
Id
Título
Views
Start time (on the stream)
End time (on the stream)
Duration
Current viewers
Id
Title
Peak viewers
Stream
Has ended
Tags
Start date and time
End date and time
Duration
has
A stream can have multiple clips
Id
Category
Name
Average viewers last 7 days
Average channels last 7 days
Rank
Number of live viewers
Number of live channels
has a
Stream has 1 category, which can be changed
Visual Paradigm Online Free Edition

4. ER

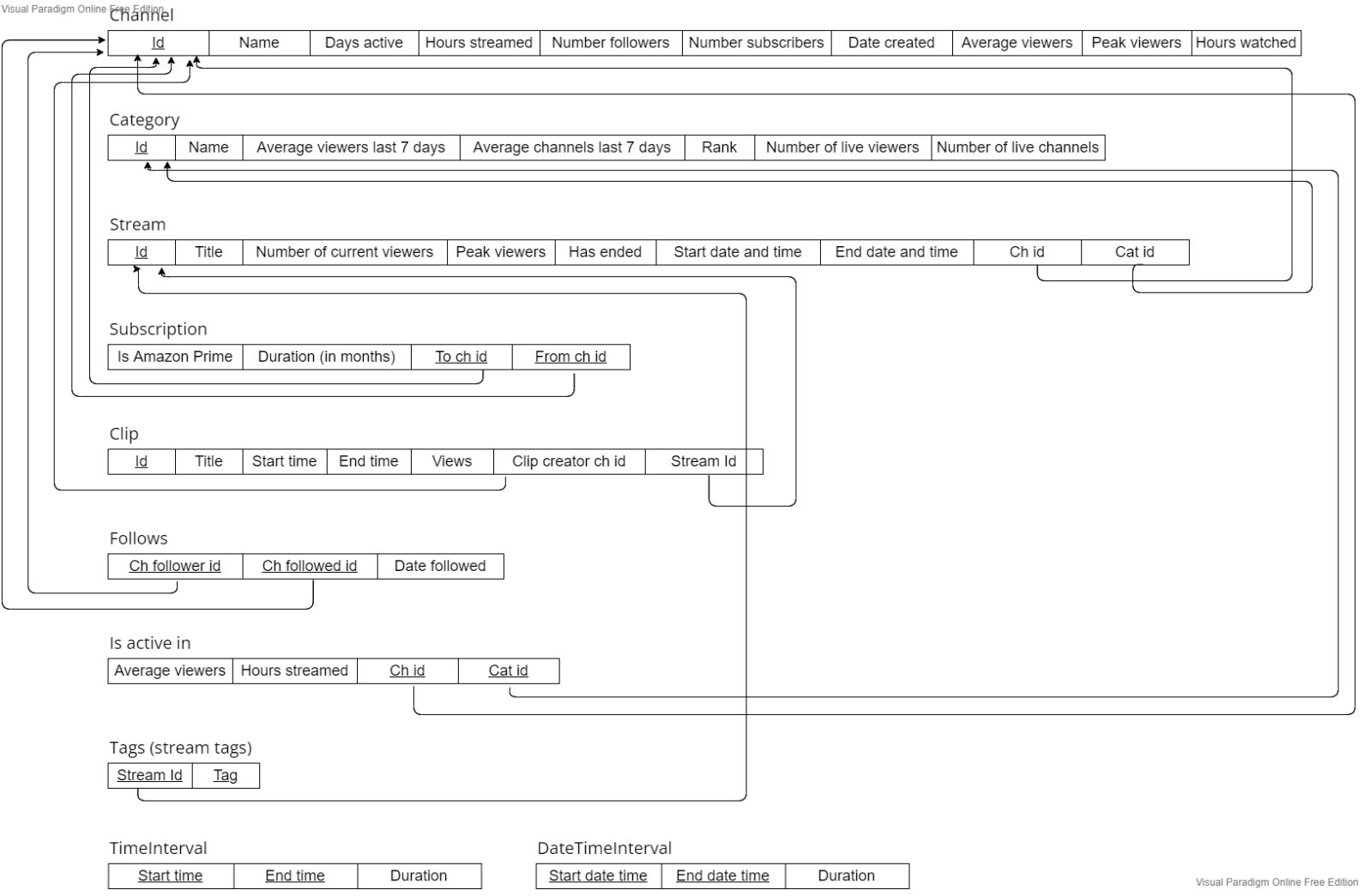

Visual Paradigm Online Free Edition
Channel
Id
Name
Days active
Hours streamed
Number followers
Number subscribers
Date created
Average viewers
Peak viewers
Hours watched
Category
Id
Name
Average viewers last 7 days
Average channels last 7 days
Rank
Number of live viewers
Number of live channels
Stream
Id
Title
Number of current viewers
Peak viewers
Duration
Has ended
Start date and time
End date and time
Ch id
Cat id
Subscription
Is Amazon Prime
Duration (in months)
To ch id
From ch id
Clip
Id
Title
Start time
End time
Duration
Views
Clip creator ch id
Stream Id
Follows
Ch follower id
Ch followed id
Date followed
Is active in
Average viewers
Hours streamed
Ch id
Cat id
Tags (stream tags)
Stream Id
Tag
Visual Paradigm Online Free Editio

5. Normalização

O ER anterior está na 2FN. Não está na 3FN porque: 1) em Stream o end date time depende totalmente de start date time e duration, e é deduzido a partir destes 2 atributos; 2) em Clip o end time depende totalmente de start time e duration, e é deduzido a partir destes 2 atributos.

Então, normalizámos a nossa database para 3FN criando mais 2 tabelas TimeInterval e DateTimeInterval, que se podem observar no ER normalizado:

6. ER normalizado 3FN



7. DDL

A seguir, criámos o DDL, ou seja, destrução e definição das tabelas.

8. DML

Algum do DML que usamos no Form (Form1.cs):

SELECT * FROM <Table>

<Table>: Twitch.STREAM, Twitch.CATEGORY, Twitch.STREAM

SELECT * FROM <Table> WHERE id=@id

SELECT Channel_name, Channel_id FROM Twitch.CHANNEL WHERE Channel_id=@Channel_id

SELECT Category_name, Category_id FROM Twitch.CATEGORY WHERE Category_id=@Category_id

SELECT * FROM Twitch.TIME_INTERVAL WHERE Start_time=@Start_time AND Duration_seconds=@Duration_seconds

SELECT * FROM Twitch.DATETIME_INTERVAL WHERE Start_date_time=@Start_date_time AND Duration_seconds=@Duration_seconds

INSERT INTO … VALUES(...)

UPDATE ... SET ... WHERE id=@id

DELETE Twitch.STREAM_TAG WHERE Stream_id = @Stream_id

DELETE Twitch.CLIP WHERE Clip_id = @Clip_id

SELECT * FROM Twitch.STREAM_TAG WHERE Stream_id=@Stream_id

"SELECT * FROM Twitch.CLIP WHERE Stream_id=@Stream_id

DELETE Twitch.FOLLOWS WHERE Follower_channel_id=@Follower_channel_id AND Followed_channel_id=@Followed_channel_id

DELETE Twitch.SUBSCRIPTION WHERE From_channel_id=@From_channel_id AND To_channel_id=@To_channel_id

9. Inserts

Introduzimos manualmente alguns tuplos nas tabelas (aproximadamente 5 por tabela).

10. UDFs

GetFollowers(@Channel_id INT) retorna os canais (Channel_name, Channel_id) que seguem o canal com id @Channel_id.

GetFollowing(@Channel_id INT) retorna os canais (Channel_name, Channel_id) que cujo o canal com id @Channel_id segue.

GetSubsFrom(@Channel_id INT) retorna os canais (Channel_name, Channel_id) a que o canal com id @Channel_id está subscrito a.

GetSubsTo(@Channel_id INT) retorna os canais (Channel_name, Channel_id) subscritos ao canal com id @Channel_id.

11. SPs

“GetStreamEndDateTime @Stream_ID INT, @EndDateTime DATETIME OUTPUT” vai à tabela DATETIME_INTERVAL buscar o end date time correto e guarda-o em @EndDateTime.

“GetClipEndTime @Clip_ID INT, @EndTime TIME OUTPUT” vai à tabela TIME_INTERVAL buscar o end time correto e guarda-o em @EndTime.

“RemoveStream @Stream_ID INT”, “RemoveChannel @Channel_ID INT” e “RemoveCategory @Category_ID INT” são 3 SPs que removem, respetivamente, a stream com o id, o canal com o id e a categoria com o id. Para isso, removem primeiro os tuplos noutras tabelas que dependem.

12. Triggers

“CREATE TRIGGER Twitch.CreateTimeInterval on Twitch.CLIP AFTER INSERT, UPDATE” adiciona, sempre que é inserido um tuplo em CLIP, um tuplo na tabela TIME_INTERVAL, correspondente ao start time e duration do clip. Tambem atualiza o tuplo existente em TIME_INTERVAL sempre que um tuplo em CLIP é alterado, pois alterar a duração ou o start time altera o end time.

“CREATE TRIGGER Twitch.CreateDateTimeInterval on Twitch.STREAM AFTER INSERT, UPDATE” faz algo semelhante, mas para as tabelas STREAM e DATETIME_INTERVAL, pois as streams têm data e hora de inicio e fim, enquanto que os clips só têm hora de inicio e fim.