Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Proyecto

Videogames and Usage/Playability Datasets

Grupo: Grupo 15
Integrantes: Felipe Alfaro
Daniel Ortuño
Vicente Pinochet
Profesores: Aidan Hogan y Sebastián Ferrada
Auxiliares: Florencia Yáñez
Fran Antonie Zautzik
Fecha de entrega: 31 de mayo de 2023

Índice de Contenidos

Índice de Figuras

1. Descripción de los conjuntos de datos

Este conjunto de datos contiene una lista de videojuegos con ventas superiores a 100,000 copias. Fue generado mediante una extracción de datos del sitio web vgchartz.com. Los datos constan de 16,598 registros con información de juegos, ordenados por cantidad de ventas ('Rank') y proporcionando información de sus ventas tanto en Japón, Norteamérica y Europa como en el resto del mundo. junto con el total de ventas. A continuación, el enlace a su respectiva página:

https://www.kaggle.com/datasets/gregorut/videogamesales

El segundo conjunto de datos consta de una lista de comportamientos de usuarios, con las columnas: 'user-id', 'game-title', 'behavior-name', 'hours'. Los comportamientos incluidos son 'compra' y 'juego'. 'Hours' indica el grado en que se realizó el comportamiento: en el caso de 'compra', el valor siempre es 1, y en el caso de 'juego', el valor representa el número de horas que el usuario ha jugado el juego.

https://www.kaggle.com/datasets/tamber/steam-video-games

2. Descripción y motivación del problema

Estos datasets ofrecen una amplia variedad de información sobre videojuegos y comportamientos de usuarios en la plataforma Steam. Esto brinda la oportunidad de explorar y analizar diferentes aspectos de la industria de los videojuegos, desde detalles de lanzamientos y calificaciones hasta patrones de compra y tiempo de juego.

A su vez los datasets ofrecen una oportunidad única para analizar los patrones de comportamiento de los usuarios y descubrir tendencias interesantes. Se podrá explorar qué juegos son más populares, qué géneros tienen más demanda, cómo influyen las críticas en las decisiones de compra y muchas otras ideas de análisis que pueden proporcionar información valiosa.

3. Opción elegida

El grupo elige una implementación web en vez de un análisis de datos, elegir una implementación web brinda la oportunidad de crear una experiencia interactiva, simple y atractiva para los usuarios y mantener la flexibilidad para futuras actualizaciones y escalabilidad.

4. Exploración inicial de los datos

- La cantidad de filas: Video Game Sales: 16.598 y Steam Video Games: 200.000
- La distribución de los atributos será en tres tablas. La primera tabla será *"GameSales"*, que contiene *Rank*, *Name*, *Platform*, *Year*, *Genre*, *P.Name*, *NA_Sales*, *EU_Sales*, *JP_Sales*, *other_Sales*, *Global_Sales*.

- La segunda tabla será *"UserBehavior"*, que posee *User_ID*, *GS.Game*, *GS.Platform*, *GS.Year*, *Behavior*, *Hours*.
- Por último, la tercera tabla será *"Publishers"*, que se define como *Name* y *Country*.

  Se pueden notar los siguientes valores relevantes según la tabla:

  – En Steam Video Games, el atributo *"Behavior"* que corresponde a si un juego sólo fue comprado (*purchase*) o si fue también fue jugado (*play*), tiene una proporción 65 %: 35 %.
  – En Video Game Sales 20 % de los juegos tienen género (*Genre*) de acción (*Action*) y 14 % de deportes (*sports*). También las plataformas (*Platform*) con mayor ocurrencia son DS y PS2, abarcando cada uno un 13 % de los datos.

  El dominio para cada atributo será:

- *Rank*: Int, *Name*: String, *Platform*: String, *Year*: Int, *Genre*: String, *P.Name*: String, *NA_Sales*: Float, *EU_Sales*: Float, *JP_Sales*: Float, *Other_Sales*: Float, *Global_Sales*: Float, para la tabla *"Games Sales"*.
- *UserID*: Int, *GS.Name*: String, *GS.Platform*: String, *Behavior*: String, *Hours*: Float, para la tabla *"UserBehavior"*.
- *Name*: String, *Country*: String, para la tabla *"Publishers"*.

  El porcentaje de datos nulos es cero.

5. Modelado Conceptual

El modelo Entidad-Relación es el siguiente:

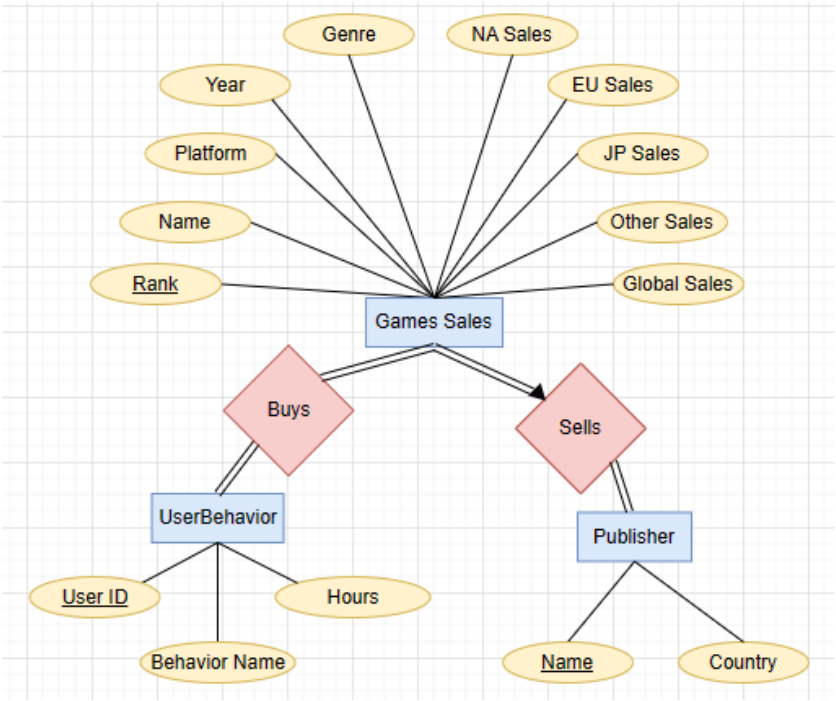

Figura 1: Modelo Entidad Relacion

Este modelo Entidad - Relación escrito como Esquema Relacional queda como sigue:

GamesSales(Rank: Int, Name: String, Platform: String, Year: Int, P.Name: String, Genre: String, NA_Sales: Float, EU_Sales: Float, JP_Sales: Float, Other_Sales: Float, Global_Sales: Float)

UserBehavior(User_ID: Int, GS.Name: String, GS.Platform: String, GS.Year: Int, Behavior: String, Hours: Float)

Publisher(Name: String, Country: String)

Buys(UB.User_ID: Int, UB.Game: String, GS.Name: String, GS.Platform: String)

*UB: UserBehavior GS: GamesSales P: Platform

En el modelo E-R, {Rank} es la llave candidata que se puso como primaria pero otra llave candidata que se usará como primaria en el Esquema Relacional porque conviene en esta distribución de los datos es {Name, Platform, Year}.

Los joins que se quieren realizar entre los datos se planean hacer de la siguiente manera:

- Es posible hacer joins entre las 2 fuentes de bases de datos que se escogieron, debido a las

similitudes en los nombres de los juegos presentes en las tablas *"GameSales"* y *"UserBehavior"* coinciden en gran medida. Se encontraron 4.340 usuarios únicos en *"UserBehavior"* (de un total de 12.393 usuarios únicos) que juegan juegos presentes en la tabla *"GameSales"*. Al realizar el join, con estos usuarios únicos se pueden alcanzar 35.040 comportamientos de juego en la tabla *"GameSales"*.

- Por otro lado, es posible hacer un join entre las tablas *"UserBehavior"* y *"Publisher"* según los nombres coincidentes en la tabla *"GameSales"*, en donde se obtiene un total de 103 publishers distintos de un total de 579.
- Finalmente, se puede hacer un join entre la tabla *"GameSales"* y *"Publisher"* con todos los publishers de la tabla *"Publisher"*, obteniendo un total de 579 publishers donde es posible relacionarlos con 16598 ventas de la tabla *"GameSales"*.

A partir del esquema generado, es posible señalar sobre su normalización que no cumple con ser 2NF, ya que en una tabla se verá que en una tabla existe un subconjunto propio de alguna llave candidata que implique atributos no primos:

- *"GameSales"*: Los atributos no primos son {P.Name, Genre, NA_Sales, EU_Sales, JP_Sales, Other_Sales, Global_Sales} pues tenemos las siguientes llaves candidatas:
  - {Rank}: No existe subconjunto propio (distinto de $\emptyset$).
  - {Name, Platform, Year}: Existe la dependencias funcional del subconjunto {Name, Platform} de esta llave candidata que sí implica {P.Name} (uno de los atributos no primos), por lo tanto, no es 2NF.
- *"Publisher"*: Los atributos no primos son únicamente {Country}, en donde su única llave candidata es {Name}. Como no existe subconjunto propio de la llave candidata, se cumple 2NF.
- *"UserBehavior"*: El único atributo no primo es {Hours}, en donde su única llave candidata es {User_Id, GS.Name, GS.platform, GS.Year, Behavior} en donde GS.Name, GS.platform, y GS.Year son la clave foránea.
  - {Hours} se puede ver que no hay dependencias funcionales de un subconjunto de esta llave candidata, ya que quedan descritas únicamente por la llave completa. Cumple 2NF.

Por no ser 2NF, no es 3NF, ahora analizamos si es Boyce-Codd:

- *"GameSales"* satisface la 1NF porque hay un valor para cada celda de las tuplas pero por la misma dependencia funcional por la que no es 2NF, no cumple con ser Boyce-Codd (Existe {Name, Platform} no superllave y deduce {P.Name}).
- En *"Publisher"* y *"UserBehavior"*, no hay más dependencias funcionales más que la llave primaria entonces se puede decir que el esquema satisface la 1NF.

6. Tratamiento de los datos

Se procesaron los datos para corregir y normalizar los siguientes errores:

- En el dataset original de *"GameSales"* faltaban las tuplas 654 y 14200, por lo que el ranking de juegos se desfasaba; se re-enumeraron.
- *"UserBehavior"* poseía una columna de puros ceros, la cual fue eliminada.
- *"UserBehavior"* tiene el nombre del juego pero para identificarlo completamente, se añadió la columna Platform *"GameSales"*, se descubrió que Year también es parte de la llave así que está por agregar.
- Cuando el usuario solo compró el juego y no lo ha jugado (*purchase*), aparecía 1.0 como las horas jugadas. Se cambió a 0.
- Se quitaron los caracteres especiales (:, ;, #, etc.) de los nombres en *"GameSales"* y *"UserBehavior"*.
- Los nombres de los juegos en las distintas bases de datos no eran iguales, ya que en una base de datos estaban con mayúsculas, y en el otro no. Para solucionar esto, se puso todo en minúscula.

6.1. Scripts SQL

Los scripts SQL utilizados para implementar la base de datos son los siguientes:

Para subir los datos al servidor se hizo uso de:

- scp -P 315 GamesSales.csv cc3201@cc3201.dcc.uchile.cl:/home/cc3201/
- scp -P 315 UserBehavior.csv cc3201@cc3201.dcc.uchile.cl:/home/cc3201/
- scp -P 315 Publisher.csv cc3201@cc3201.dcc.uchile.cl:/home/cc3201/

Para crear las tablas se usó:

```
CREATE TABLE GamesSales (
    rank INTEGER,
    name VARCHAR(255),
    platform VARCHAR(255),
    year INTEGER,
    genre VARCHAR(255),
    publisher VARCHAR(255),
    na_sales FLOAT,
    eu_sales FLOAT,
    jp_sales FLOAT,
    other_sales FLOAT,
    global_sales FLOAT,
    PRIMARY KEY (name, platform, year),
    FOREIGN KEY (publisher) REFERENCES Publisher(name)
);
```

```sql
CREATE TABLE UserBehavior (
    userid INTEGER,
    game VARCHAR(255),
    platform VARCHAR(255),
    behavior VARCHAR(255),
    hours FLOAT,
    PRIMARY KEY (userid, game, platform, year, behavior),
    FOREIGN KEY (game, platform, year) REFERENCES GamesSales(name, platform, year)
);
```

```sql
CREATE TABLE Publisher (
    name VARCHAR(255) PRIMARY KEY,
    country VARCHAR(255)
);
```

Por último, para rellenar estas tablas, se utilizó:

```sql
COPY GamesSales FROM '/home/cc3201/GamesSales.csv' DELIMITER ',' CSV HEADER;
```

```sql
COPY UserBehavior FROM '/home/cc3201/UserBehavior.csv' DELIMITER ',' CSV HEADER;
```

```sql
COPY Publisher FROM '/home/cc3201/Publisher.csv' DELIMITER ',' CSV HEADER;
```

7. Descripción del estado final de la base de datos

Al cargar los datos arreglados a la base de datos, se obtendrán las siguientes tablas:

- Para *"GameSales"* se tendrán las siguientes columnas: *Rank*, *Name*, *Platform*, *Year*, *P.Name*, *Genre*, *NA_Sales*, *EU_Sales*, *JP_Sales*, *Other_Sales*, *Global_Sales*, todos con 16598 datos.
- Para *"Publisher"* se tendrán las siguientes columnas: *Name* y *Country* con 121 datos.
- Para *"UserBehavior"* se tendrán las siguientes columnas: *UserID*, *GS.Name*, *GS.Platform*, *GS.Year*, *Hours*, con 207083 datos.

Durante la carga de los datos existieron múltiples problemas. Existió un problema a la hora de crear las tablas, ya que aparecían tuplas repetidas, las cuales se supone fueron eliminadas durante el tratamiento de los datos. Además, cuando se señalaron las llaves primarias, ocurrió un error que nos señalaba que se duplicaban tuplas en la tabla *"Publisher"*. Otro problema, fue la existencia de caracteres especiales ocultos, la presencia de comas, y por último, espacios indeseados en algunos atributos.

El estado final de los datos es el siguiente:

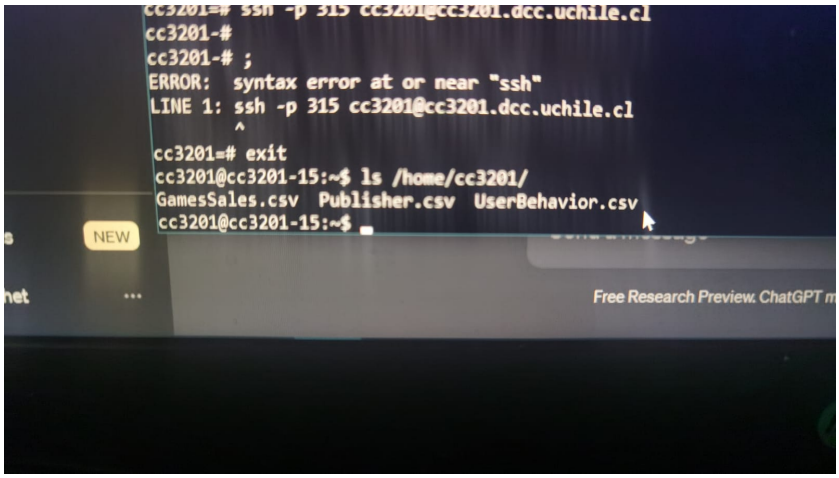

Figura 2: Conjuntos de datos subidos al servidor del curso

8. Consultas con parámetros

Consulta de ventas globales por género:
¿Cuál es el total de ventas globales para el género [género]?

```
SELECT Genre, SUM(Global_Sales)
FROM GamesSales
GROUP BY Genre;
```

Consulta de juegos más populares de un editor:
¿Cuáles son los juegos más populares publicados por [Publisher]?

```
SELECT Rank, Name, Global_Sales
FROM GamesSales
WHERE publisher LIKE 'nintendo';
FETCH FIRST 5 ROWS ONLY;
```

Consulta de horas de juego por usuario y género:
¿Cuántas horas ha jugado el usuario [UserID] en juegos del género [género]?

```
SELECT Hours
FROM UserBehavior
WHERE game LIKE 'spore';
```

Consulta que juegos tiene el menor tiempo de juego por usuario y mayor cantidad de ventas:
¿Cuál es el juego que tiene menos tiempo de juego que [Hours] y mayor cantidad de ventas que [Global_Sales]?

```
SELECT Name, Hours, Global_Sales
```

FROM UserBehavior
WHERE Behavior = "purchase"AND Hours < ;

Consulta que juegos tiene el mayor tiempo de juego por usuario y menor cantidad de ventas:
¿Cuál es el juego que tiene mayor tiempo de juego que [Hours] y menor cantidad de ventas que [Global_Sales]?

SELECT Name, Hours, Global_Sales
FROM UserBehavior
WHERE Behavior = "purchase"AND Hours > ;

Consulta cuál es el país de publisher que tiene más ventas:
¿Cuál es el país de [Publisher] con mayor ventas?

SELECT Global_Sales
FROM { SELECT Country, SUM(Global_Sales)
FROM Games_Sales, Publisher
WHERE Country =
WHERE Publisher = ;

9. Optimización de la base de datos

Para garantizar respuestas rápidas por parte de la base de datos, se hizo uso de solo un índice (para no generar tanto uso de memoria) en la tabla de *"GameSales"* específicamente en los atributos de nombre del *"Publisher"* y nombre del juego, ya que estos son los valores que mayor uso se prevé que tendrán:

```
CREATE INDEX names ON GamesSales (name, P.name);
```

Debido a complicaciones inesperadas al visualizar correctamente los datos en SQL, no se pudieron obtener las capturas de pantalla que comparasen la eficiencia de las consultas con y sin índices pero esta es la consulta que se realizará una vez estén bien definidos los datos:

```
EXPLAIN ANALYZE
CREATE MATERIALIZED VIEW HoursPerGame AS
  SELECT name, AVG(hours) AS avg_hours
  FROM GamesSales, UserBehavior
  WHERE name = GS.Name
  GROUP BY Name;
```

Con esta vista se podrá visualizar los promedios de horas para cada juego y tenerlos más a mano en vez de estar ejecutando la consulta cada vez. En la eventual actualización de la base de datos, bastará hacer una actualización con el comando:

```
REFRESH MATERIALIZED VIEW HoursPerGame;
```