

**Projecte de base de dades**

***ClashSayale 2021-2022 – Fase 2***

**Llistat de membres (nom i correu):**

Marc Geremias - [marc.geremias@students.salle.url.edu](mailto:marc.geremias@students.salle.url.edu)

Irina Aynés - [irina.aynes@students.salle.url.edu](mailto:irina.aynes@students.salle.url.edu)

Marc Valsells - [marc.valsells@salle.url.edu](mailto:marc.valsells@salle.url.edu)

Albert Tomàs - [albert.tomas@students.salle.url.edu](mailto:albert.tomas@students.salle.url.edu)

**Data d'entrega:** 14/01/2022

## Taula de continguts

<b>1</b>	<b>INTRODUCCIÓ .....</b>	<b>3</b>
<b>2</b>	<b>ACTUALITZACIÓ DEL MODEL ENTITAT-RELACIÓ.....</b>	<b>4</b>
<b>3</b>	<b>ACTUALITZACIÓ DEL MODEL RELACIONAL .....</b>	<b>6</b>
<b>4</b>	<b>SELECCIÓ DEL TIPUS DE DADES.....</b>	<b>8</b>
<b>5</b>	<b>CODIFICACIÓ DEL MODEL FÍSIC.....</b>	<b>11</b>
<b>6</b>	<b>IMPORTACIÓ DE LA BASE DE DADES .....</b>	<b>12</b>
	ARENA_PACK.CSV .....	12
	ARENAS.CSV .....	12
	BATTLES.CSV .....	12
	BUILDINGS.CSV .....	13
	CARDS.CSV .....	13
	CLAN_BATTLES.CSV .....	14
	CLAN_TECH_STRUCTURES.CSV .....	14
	CLANS.CSV .....	14
	FRIENDS.CSV .....	14
	MESSAGES_BETWEEN_PLAYERS.CSV .....	15
	MESSAGES_TO_CLANS.CSV .....	15
	PLAYER_PURCHASES.CSV .....	16
	PLAYERS.CSV .....	16
	PLAYERS_ACHIEVEMENTS.CSV .....	16
	PLAYERS_BADGE.CSV .....	17
	PLAYERSCARDS.CSV .....	17
	PLAYERS_CLANS.CSV .....	17
	PLAYERS_CLANS_DONATIONS.CSV .....	17
	PLAYERS_DECKS.CSV .....	18
	PLAYERS_QUESTS.CSV .....	18
	SEASONS.CSV .....	19
	SHARED_DECKS.CSV .....	19
	TECHNOLOGIES.CSV .....	19
<b>7</b>	<b>VALIDACIÓ DE LA BASE DE DADES.....</b>	<b>20</b>
	JUGADORS .....	20
	CARTES .....	22
	CLANS .....	23
	BATALLES.....	26
	ARENES .....	28
	MISSIONS.....	30
	ASSOLIMENTS .....	32
	BOTIGA – PAQUETS D’ARENA, OFERTES, COFRES I EMOTICONES .....	33
<b>8</b>	<b>CONCLUSIONS .....</b>	<b>37</b>
	RECURSOS EMPRATS.....	37
	LLIÇONS APRESES I CONCLUSIONS .....	38

# 1 Introducció

La creació d'un model físic no és una tasca de caire molt complicat, a vegades salten errors que costa veure, però amb constància i, sobretot, entenent el que fas i havent estudiat la teoria, es pot realitzar un model físic amb condicions.

Aquesta segona fase de la pràctica, consistia en passar el nostre model entitat-relació i model relacional a ja un model físic codificat. Els objectius d'aquesta fase eren intentar ja aconseguir una base de dades funcional i programada que, al cap i a la fi, son les que s'utilitzen en el món professional.

Per fer-ho, ha calgut seguir les classes de model físic implementades abans de nadal i també ha estat molt útil fer les activitats d'avaluació continua ja que ajudaven molt a acostumar-se en l'entorn SQL i com funciona el llenguatge, la creació i destrucció de taules, la importació d'atributs a les taules i els tipus de variables, i la validació per comprovar que aquestes taules han estat ben omplertes.

També un altre objectiu era familiaritzar-se amb l'entorn .csv, sobretot en com importar dades d'un fitxer extern, buscar el path correcte per a què el codi funcioni i el treball amb fitxers en general.

Com a últim objectiu, tenim el mateix que en cada fase, que és el treball en equip. Al cap i a la fi, aquest objectiu és dels més importants ja que en el món laboral, és molt probable que se'ns faci treballar en grups i s'ha de saber conviure i fer projectes en grups. En aquest projecte, el més important és dividir-se bé la feina, saber què fa cadascú i sobretot, que tots els membres del grup posin de la seva part.

Els rols que s'han pres en aquesta fase han estat molt equilibrats, sempre s'ha intentat dividir tota la feina de manera equitativa entre tots els membres del grup. Ens vam repartir unes quantes taules per cada membre i vam fer els `CREATE`s de cada una (basant-nos en el nostre model relacional, òbviament). Després ens vam repartir unes taules diferents a les que havíem creat cadascú de manera que quan féssim els `IMPORTS` a cada taula poguéssim veure altres parts que no sabíem, d'aquesta manera hem aconseguit conèixer el projecte en tota la totalitat cada membre del grup.

En conclusió, ara es proporcionaran explicacions molt més detallades de cada taula i cada importació dels fitxers csv, així com els atributs i els tipus de dades que s'utilitzaran. També explicarem els canvis i modificacions que hem realitzat als models entitat-relació i relacional ja que després de l'entrevista i un cop hem revisat els fitxers que se'ns han proporcionat, hem vist que potser ens faltaven atributs, o alguna relació no estava del tot bé, o ens sobrava alguna taula i ens faltava una altra, etc...

Per tant, ara començarem explicant l'actualització dels dos models i les correccions que s'han fet.

## 2 Actualització del model entitat-relació

Al llarg de fer la implementació física es van haver de modificar diferents entitats del model conceptual per poder importar correctament la informació dels fitxers csv. A continuació deixem les explicacions de les diferents modificacions que hem fet i, tot seguit la captura del nou model conceptual en horitzontal per a què es pugui apreciar millor:

- La primera modificació que es va haver de fer va consistir en afegir diversos atributs a l'entitat generalitzada "Millora" per així guardar tota la informació de cada tecnologia i estructura.
- A més dels atributs de les millores també es va haver de canviar les relacions reflexives d'estructures i tecnologies ja que el prerrequisit d'obtenir una nova estructura sempre venia donada per una altre estructura i el mateix passava amb les tecnologies, per això es van fer 2 reflexives separades. I junt amb amb aquesta modificació es va haver d'afegir l'atribut "nivell\_prerrequisit" a l'entitat requereix\_tecnologia.
- També, per poder saber quina tecnologia i quina estructura té cada clan es van crear dos relacions N:M anomenades tenen\_estructura i tenen\_tecnologia per així guardar la data, el nivell i el clan el qual té aquesta millora.
- Una altre cosa que vam canviar va ser que la relació de Jugador amb Batalla ara disposa de dues relacions en comptes d'una, i aquestes es diuen: Guanya i Perd, les quals tenen un atribut de la relació anomenat num\_trofeus, que són els trofeus que es sumen al guanyador i els que es resten al perdedor. També vam afegir la entitat Insígnia al guanyador ja que un Jugador aconsegueix una insígnia en cas que guanyi la batalla.
- Anteriorment, teníem que la entitat Missió era una generalització de les entitats Insígnia i Assoliment, però ho vam acabar canviant i les vam separar. Ara, l'entitat missió surt d'una relació amb Jugador, Missió i Arena ja que una missió és aconseguida per un jugador en una arena determinada. També té atributs a la relació "completen", que són l'or i l'experiència guanyada per completar aquella missió. Finalment, cal recalcar que ara també l'entitat Missió té una relació reflexiva amb si mateixa que s'anomena "depen" ja que l'enunciat deixa ben clar que una missió depèn d'una altra missió.
- Posteriorment, es va afegir la entitat targeta de crèdit per a guardar la informació d'aquesta i poder-la relacionar amb la compra d'un article que fa un jugador. Al establir aquesta nova entitat, es va modificar la relació compren i va fer una relació ternària N:M:P entre targeta de crèdit, jugador i article.
- Per a poder relacionar la raresa d'una cartes s'ha afegit una relació entre el tipus de raresa d'una carta amb les cartes que pot haver-hi en un cofre. Aquesta relació es 1:N.
- Després, per afegir la quantitat d'or que hi ha en un paquet d'arena, ens vam donar compte que ve donat per l'arena. Per tant es va afegir una relació entre paquet d'arena i arena on inclou aquest atribut d'or.
- Finalment, es va afegir una nova relació anomenada amics, per explicar que els jugadors poden tenir amics. Aquesta relació es N:M, ja que cada jugador pot tenir varis amics.



### **3 Actualització del model relacional**

Mentre s'anava actualitzant el model conceptual s'ha anat actualitzant parlament el model conceptual sempre tenint en compte les noves taules que s'havien d'afegir o treure segons si s'havia canviat el tipus de relació (quan s'ha passar de 1:N o de N:M).

Algunes taules han estat afegides quan la multiplicitat ha canviat de 1:N a N:M i altres eliminades per la mateixa raó però passant de N:M a 1:N.

A la següent pàgina deixem la imatge del nou model relacional modificat:



## 4 Selecció del tipus de dades

Bé, el nostre model físic l'hem compost de tres fitxers .sql per tal de mantenir el codi el màxim ordenat possible (GB5\_modelfisic.sql, GB5\_importacio.sql i GB5\_validacio.sql). Després S'ha fet ús dels 24 fitxers .csv que se'ns han proporcionat per fer aquesta fase dos del projecte. Finalment, hem usat les nostres 50 taules que s'han creat a GB5\_modelfisic.sql, les quals provenen del nostre model relacional entregat a la fase 1 i que, com s'ha vist en apartats anteriors, ha estat modificat i arreglat per a què es pugui guardar i importar les dades de la manera més semblant als .csv possible.

A continuació explicarem algunes taules que vam crear al fitxer GB5\_modelfisic.sql i els seus atributs, així com les decisions preses a l'hora de posar un tipus determinat per a cada atribut:

JUGADOR:

```
-- Taula Jugador
CREATE TABLE Jugador(
    tag_jugador VARCHAR (255),
    nom VARCHAR(255),
    experiencia INTEGER,
    trofeus INTEGER,
    targeta_credit VARCHAR(255),
    PRIMARY KEY (tag_jugador),
    FOREIGN KEY (targeta_credit) REFERENCES targeta_credit(numero) ON DELETE CASCADE
);
```

Aquesta taula “Jugador”, té 5 atributs entre els quals trobem: tag\_jugador (hem decidit que sigui un VARCHAR ja que era una combinació de lletres i números), nom (que és un VARCHAR perquè es tracta d'una cadena de caràcters), experiencia (que es tracta d'un INTEGER ja que és un nombre), trofeus (és INTEGER per la mateixa raó que “experiència”) i targeta\_credit (que es tracta d'un VARCHAR ja que es tracta d'una combinació de nombres massa llarga per omplir un INTEGER). Després s'ha posat el tag\_jugador com a clau primària i targeta\_credit com a clau forània ja que ve de l'entitat pròpia targeta\_credit.



## BATALLA:

```
-- Creació de la taula Batallen
CREATE TABLE Batalla (
    ID_batalla SERIAL,
    data DATE,
    durada TIME,
    ID_temporada VARCHAR(255),
    ID_arena INTEGER,
    clan_battle INTEGER,
    PRIMARY KEY (ID_batalla),
    FOREIGN KEY (ID_temporada) REFERENCES Temporada (ID_temporada) ON DELETE CASCADE,
    FOREIGN KEY (ID_arena) REFERENCES Arena (ID_arena) ON DELETE CASCADE
);
```

Aquesta taula “Batalla”, té 6 atributs entre els quals trobem: ID\_batalla (hem decidit que sigui un SERIAL ja que aquest id no el trobem en cap .csv i l’hem hagut d’inventar nosaltres), data (que es tracta d’un DATE ja que està escrit en aquest format (YYYY/MM/DD)), durada (que és de tipus TIME ja que està escrit d’aquesta forma (HH:MM:SS)), ID\_temporada (que és un VARCHAR perquè es tracta d’una combinació de lletres i números), ID\_arena (que es tracta d’un INTEGER ja que és un nombre) i clan\_battle (és INTEGER per la mateixa raó que “experiència”). Després s’ha posat el ID\_batalla com a clau primària i ID\_temporada i ID\_arena com a claus foranies ja que venen de les entitats pròpies Temporada i Arena, respectivament.

## CARTA:

```
-- Creació de la taula Carta
CREATE TABLE Carta (
    nom VARCHAR (255),
    dany INTEGER,
    velocitat_atac INTEGER,
    raresa VARCHAR(255),
    arena INTEGER,
    PRIMARY KEY (nom),
    FOREIGN KEY (raresa) REFERENCES Raresa (nom) ON DELETE CASCADE,
    FOREIGN KEY (arena) REFERENCES Arena(ID_arena) ON DELETE CASCADE
);
```

Aquesta taula “Carta”, té 5 atributs entre els quals trobem: nom (hem decidit que sigui un VARCHAR ja que es tracta d’una cadena de caràcters), dany (que es tracta d’un INTEGER ja que és un nombre), velocitat\_atac (és INTEGER per la mateixa raó que “experiència”) i raresa (que es tracta d’un VARCHAR ja que es tracta d’una cadena de caràcters) i arena (que es tracta d’un INTEGER ja que és un número. Després s’ha posat el nom com a clau primària i raresa i arena com a claus foranies ja que venen de les entitats pròpies Raresa i Arena, respectivament.

COFRE:

```
CREATE TABLE Cofre (  
    ID_cofre INTEGER,  
    nom_cofre VARCHAR(255),  
    quantitat_cartes INTEGER,  
    raresa VARCHAR(255),  
    Temps INTEGER,  
    PRIMARY KEY (ID_cofre),  
    FOREIGN KEY (ID_cofre) REFERENCES Article(ID_article) ON DELETE CASCADE,  
    FOREIGN KEY (raresa) REFERENCES Raresa(nom) ON DELETE CASCADE  
);
```

Aquesta taula correspon a l'entitat "Cofre" on s'hi troba 5 atributs. Aquests són ID\_Cofre (és un número enter ja que ve donat per ID\_article, per tant és un INTEGER), nom\_cofre (establert com un VARCHAR ja que és una cadena de caràcters), quantitat\_cartes (és el número en enters de cartes que hi ha al cofre, per això correspon a un INTEGER), raresa (és una cadena de caràcters que estableix el nom de la raresa, per aquesta raó s'ha posat un VARCHAR) i el temps (és quantitat de temps que tarda el cofre per desbloquejar-se per tant és un número enter i per això s'ha posat INTEGER). Després s'ha afegit com a clau primària ID\_cofre i com a foranes ID\_cofre (ja que és una generalització d'article) i raresa (ve referència per l'entitat raresa, ja que és qui ens proporciona el nom de la raresa de la nostra carta del cofre).

## 5 Codificació del model físic

A partir del model relacional i de la selecció de tipus de dades realitzat anteriorment, cada membre de l'equip ha creat les taules de la part del model adjudicades.

Primer s'han afegit les taules de cada entitat. Per a crear aquestes taules, s'ha utilitzat la comanda CREATE TABLE, en aquesta hem afegit els atributs que teníem establerts en el model relacional i al costat de cada atribut s'ha afegit el tipus, seleccionat anteriorment.

Posteriorment, s'han determinat les restriccions adequades per a cada taula.

I finalment s'han anat col·locant en ordre de ús, és a dir, per a que les restriccions de claus foranes en referència a altres taules poguessin ser correctes s'ha posat la taula que conte la clau forana referenciada amb una altra entitat a baix i la taula de l'altre entitat amunt.

A continuació és mostra un exemple de com s'han implementat les taules i el seu ordre:

```
-- Taula Targeta de credit
CREATE TABLE targeta_credit(
    numero VARCHAR (255),
    caducitat DATE,
    PRIMARY KEY (numero)
);

-- Taula Jugador
CREATE TABLE Jugador(
    tag_jugador VARCHAR (255),
    nom VARCHAR(255),
    experiencia INTEGER,
    trofeus INTEGER,
    targeta_credit VARCHAR(255),
    PRIMARY KEY (tag_jugador),
    FOREIGN KEY (targeta_credit) REFERENCES targeta_credit(numero) ON DELETE CASCADE
);
```

Com es pot observar a la imatge, l'entitat jugador te una clau forana referenciada a l'entitat "Targeta\_credit", i com s'ha explicat anteriorment, per a poder agafar la clau referenciada s'ha col·locat l'entitat "targeta\_credit" abans que l'entitat "Jugador" (entitat que conte la clau forana referenciada a "targeta\_credit").

També es pot observar la creació de taules amb el CREATE TABLE , el tipus establert i les restriccions aplicades.

Finalment, es va afegir la comanda DROP TABLE per a cada entitat a la part superior de l'script anterior a les taules, d'aquesta manera cada cop que executem l'script es creen taules noves i es boren les anteriors (si existeixen).

## 6 Importació de la base de dades

### *Arena\_pack.csv*

En aquest primer csv només hi ha tres columnes a inserir les quals es troben en una sola taula arena\_pack\_arena. Tot i això no s'han pogut inserir les dades directament en aquesta taula donat que alguns ids d'arena pack del csv arena\_pack no es troben en el playerpurchase.csv. Per tant prèviament s'ha hagut d'inserir les dades dels articles i llavors fer el COPY del csv en una taula temporal. A l'hora de fer el INSERT a la taula d'arena\_pack\_arena s'ha agafat la taula temporal i s'ha fet un JOIN amb la taula arena\_pack igualant els dos ids d'arena\_pack. Si algun id només estava present en una de les dues taules aquella fila no s'ha inserit donat que no seria satisfer la relació de la clau forania.

### *Arenas.csv*

Per poder importar tot el csv en relació a arenes primer s'ha creat una taula en el model físic anomenada "arena" aquesta taula s'encarrega de guardar tota d'informació de cada arena, per fer-ho s'han posat les columnes corresponents als atributs necessaris a guardar els quals són: id\_arena, títol, nombre\_min i nombre\_max. Tots aquests atributs són de tipus INTEGER menys el títol el qual és un VARCHAR(255) que ens ajuda a guardar el nom de cada arena.

Com que la taula del model físic correspon perfectament amb el csv s'ha pogut fer una importació molt senzilla la qual consisteix en utilitzar la funció "COPY" per copiar la informació del .csv a la taula corresponent d'arenes.

### *Battles.csv*

Per importar aquest fitxer .csv primer s'ha creat una taula temporal auxiliar que s'anomena "battle" en la qual inserirem tots els atributs i els seus respectius tipus que trobem al fitxer proporcionat: winner (integer), loser (integer), winner\_score (integer), loser\_score (integer), date (date), duration (time), clan\_battle (integer), aquest últim atribut, l'afegirem més endavant a la taula "lluiten" que està relacionada amb el fitxer "clans\_battles.csv", ho veurem en l'explicació d'aquest. Un cop inserits en aquesta taula temporal, enviem els atributs a tres taules diferents (com ho hem fet nosaltres al model relacional) que són: batalla, guanya i perd. A batalla simplement hem afegit la data i la durada. A guanya hem afegit el tag del jugador que guanya la batalla i el nombre de trofeus que li suma, i a perd, el tag del jugador que perd la batalla i el nombre de trofeus que li resta.

Cal recalcar que per fer això, sobretot amb les taules "guanya" i "perd", s'ha hagut de fer un SELECT dins d'un altre SELECT per poder accedir al tag del jugador ja que només volíem agafar el tag del jugador que complís que la pila relacionada amb aquell jugador fos la pila del que havia guanyat la batalla i el mateix amb el jugador que perd però agafant la pila relacionada amb el que perdia la batalla.

Un cop finalitzada la importació, s'ha destruït la taula temporal "battle" amb un DROP TABLE battle i també hem fet el drop table de playersdeck just per sobre, que era una altra taula que havíem creat anteriorment per afegir la pila però que no podíem esborrar ja que necessitàvem la informació per fer aquesta importació.

### ***Buildings.csv***

Per importar aquest fitxer primer s'ha creat una taula temporal auxiliar anomenada “buildings” amb tots els atributs necessaris per importar tota la informació del fitxer buildings.csv, seguidament es fa una còpia dels atributs (nom\_millora, descripcio, cost, mod\_damage, mod\_hit\_speed, mod\_radius, mod\_spawn\_damage, mod\_lifetime) de la taula “buildings” a la taula millora.

Més endavant també es fa una còpia dels atributs (id\_estructura, minim\_trofeus) de la taula “buildings” a la taula “estructura” i finalment es fa una última còpia desde la taula “buildings” a la taula “requereix\_estructura” on es guarden els atributs (id\_estructura\_nova, id\_estructura\_requerida).

Un cop copiada tota la informació de la taula auxiliar es fa un DROP TABLE IF EXISTS de la taula “buildings”.

### ***Cards.csv***

Per a importar aquest csv, primer s'ha revisat quines dades incloïa per a saber a quines taules importar-les. Un cop trobades les taules, s'ha creat una taula temporal introduint totes les dades que portava, a mesura que hem entrat les dades, s'ha anat decidint el tipus (varchar, float, integer, date...).

Posteriorment, s'han copiat les dades del csv donat de la carpeta datasets, a la taula temporal que s'acabava de crear.

Aquest csv, inclou atributs de les entitats, Cartes, Raresa, Edifici, Tropa, Encanteri i arena.

Per a introduir els atributs del csv a les entitats comentades anteriorment s'ha utilitzat la consulta “Insert into select”. Per a fer-ho, s'ha seleccionat els atributs de les nostres taules on es volia introduir les dades en el insert into, després en el select s'ha seleccionat els atributs que es volien agafar de la taula temporal i finalment s'ha posat from per determinar de quina taula venien, en aquest cas s'ha posat la taula temporal que s'ha creat anteriorment.

En la entitat carta, s'han importat els atributs nom, danys i velocitat d'atac. Per a l'entitat Raresa, s'ha afegit la seva PK, el nom. A l'entitat Edifici, s'ha afegit la seva vida, després a l'entitat Tropa s'ha inserit el dany d'aparició, posteriorment a l'entitat Encanteri, s'ha importat el radi i finalment en l'entitat Arena la seva PK.

Per aquelles entitats on la PK no estava donada pel csv s'ha afegit al model físic un camp auto incremental anomenat “Serial”, d'aquesta manera cada cop que s'importava una dada, s'afegia un numero com a ID d'aquella entitat.

Finalment s'han afegit al inici les comandes “DROP TABLE” i “DELETE TABLE” per eliminar la taula temporal creada i la nostra taula (si ja existeixen).

### ***Clan\_battles.csv***

Per a fer la importació d'aquest .csv, primer de tot, com a la resta d'importacions, hem creat una taula temporal on poder copiar totes les dades del fitxer csv en ella anomenada "clans\_battle". El csv en concret contenia quatre atributs: battle, clan, start\_date, end\_date. S'han copiat aquests atributs a la taula temporal creada anteriorment de la carpeta de datasets, que és on tenim tots els fitxers csv.

Un cop fet això s'han inserit totes les dades de la taula temporal a la taula que s'ha creat al model físic la qual hem anomenat "lluïten" que és com relacionàvem clans amb batalles al model relacional. Aquesta taula te quatre atributs també, i son els mateixos: id\_batalla, tag\_clan, data\_inici, data\_fi.

Simplement s'ha hagut de fer un "INSERT INTO lluïten" dels atributs de la taula que havíem creat per tal d'omplir la taula que ens interessa per guardar la nostra base de dades. El id\_batalla per això, s'ha extret del csv "battles" ja que hi havia un atribut que com hem comentat en el seu respectiu apartat, es deia clan\_battle i posava el id de la batalla de clan. De manera que s'ha fet un altre SELECT per poder obtenir aquest id\_batalla.

Finalment, s'ha fet un "DROP TABLE clans\_battle" per eliminar aquesta taula temporal.

### ***Clan\_tech\_structures.csv***

Per importar la informació del fitxer clan\_tech\_structures.csv primer s'ha creat una taula temporal auxiliar anomenada "clans\_tech\_structures", aquesta taula ens serveix per fer els imports a les taules tenen\_tecnologia i tenen\_estructura d'una manera molt més senzilla.

Un cop tota la informació del fitxer csv s'ha importat a la taula auxiliar s'ha pogut començar amb la importació a les taules del nostre model físic. A cada fila del csv tenim el tag del clan, el nom de la tecnologia, el nom de l'estructura, la data de quan es va obtenir aquesta tecnologia/estructura i el nivell al qual tenen cada millora.

Després d'analitzar el csv ens hem adonar que mai coincidien les files de noms de tecnologies amb les files de noms de les estructures, de manera que hem entès que quan hi havia el nom de la tecnologia i el nom de l'estructura estava en NULL s'estava afegint una tecnologia i viceversa.

Així doncs per importar correctament la informació de la taula temporal a les taules "tenen\_tecnologia" i "tenen\_estructura" hem ficat una condició en el SELECT del INSERT INTO el qual demana que el nom que s'està afegint no pot ser NULL, d'aquesta manera aconseguim fer l'importació a les dues taules que guarden que té cada clan

### ***Clans.csv***

Per importar el csv amb la informació de clans no es va haver de crear cap taula auxiliar ja que les columnes del fitxer a importar coincidien amb la informació a afegir de la taula clan de manera que es va fer un "COPY clan" de tota la informació dels clans als atributs (tag\_clan, nom, descripcio, trofeus\_minims, nombre\_trofeus, puntuacio) de la taula clan.

A més l'atribut "tag\_clan" el vam haver de canviar del tipus "INTEGER" al tipus "VARCHAR(255)", ja que el seu contingut estava format per números i lletres.

### ***Friends.csv***

Per a importar aquest csv, s'ha creat una taula temporal introduint totes les dades que portava, a mesura que hem entrat les dades, s'ha anat decidint el tipus (varchar, float, integer, date....).

Posteriorment, s'han copiat les dades del csv donat de la carpeta datasets, a la taula temporal que s'acabava de crear.

Aquest csv inclou simplement dos atributs: C1 i C2, els quals fan referència al ID del jugador que demana sol·licitud i el ID del jugador que la rep.

Després de copiar les dades del csv a la taula temporal que hem creat, s'ha fet un "INSERT INTO amics" (que és la taula permanent que s'ha creat al model físic i també, com es pot observar en el relacional té dos atributs que son tag\_jugador1 i tag\_jugador2) per tal d'omplir la taula.

Finalment s'ha afegit la comanda "DROP TABLE" per eliminar la taula temporal creada.

### ***Messages\_between\_players.csv***

Per importar el contingut de l'arxiu messages\_between\_players.csv primer s'ha creat una taula temporal amb el contingut d'aquest donat que no tots els atributs dels csv van a la mateixa taula, aquesta taula tenia les següents columnes: id INTEGER, sender VARCHAR(255), receiver VARCHAR(255), text TEXT, date DATE, answer INTEGER, total INTEGER.

A continuació aquest contingut ha sigut importat en dues taules diferents. El id del missatge, el cos, la data i el id del missatge resposta s'han importat a la taula missatge que és en la que es guarda tota la informació dels missatges en si, ha sigut necessari afegir un WHERE, per tal de que verifiqués que el id, el text del missatge i la data no siguin NULL, donat que el csv proporcionat té una línia en blanc al final la qual és interpretada com una fila amb el valor NULL en totes les seves columnes i d'aquesta manera la podem obviar.

Per acabar s'ha inserit el id del missatge, el tag del jugador que envia el missatge i el tag del jugador que rep el missatge a la taula conversen per tal d'evitar la fila en blanc mencionada anteriorment també s'ha afegit un WHERE per comprovar que sigui una fila amb dades vàlides.

### ***Messages\_to\_clans.csv***

Hi ha part de la informació de messages\_to\_clans.csv que és compartida amb el punt anterior, específicament el id del missatge, el text, la data i la resposta, aquesta informació també s'ha guardat en la taula de missatge. Abans d'inserir aquesta informació a la taula s'ha creat una taula temporal per facilitar la importació. S'ha trobat en que el id d'aquest csv tornava a començar des de 1 i per tant ja no era vàlid per utilitzar-lo com a id de la taula missatge.

Per solucionar-ho s'ha fet un COUNT(id) de la taula de missatges i aquest s'ha guardat en la taula temporal creada pels missatges dels jugadors amb id -2147483648 (difícilment es trobarà un missatge amb aquest id) i el valor del COUNT en la columna answer. A continuació, a l'hora d'inserir el id de missatge i de la resposta del messages\_to\_clan.csv s'ha afegit el valor del COUNT en el id del missatge i en el id de la resposta, per obtenir el valor a inserir de la columna id en el select de de la taula temporal s'ha fet la següent consulta: (id + (SELECT answer FROM msgPlayersTmp WHERE id = -2147483648)). D'aquesta manera el COUNT guardat prèviament és suma al id, en el cas de la columna de la resposta s'ha utilitzat exactament el mateix mètode.

Un cop s'han inserit les dades a la taula missatge aquest ja és pot relacionar amb el jugador i el clan. S'han tornat a agafar els valors de la taula temporal: id (sumant-li el COUNT de nou), tag del jugador i tag del clan; i aquests han sigut inserits a la taula Envia, d'aquesta manera ja s'ha fet la relació trinària.

### ***Player\_purchases.csv***

A la hora d'importar el csv de "Player\_purchase", primer es van mirar els atributs que conte per saber on importar cada dada. Després es van establir els tipus de cada atribut i es va crear una taula temporal anomenada "player\_purchase". Per a poder copiar les dades a les nostres taules, es va utilitzar la consulta "COPY", es van copiar les dades del csv de "player\_purchase" que teníem al datasets a la taula temporal que acabàvem de crear anomenada també "player\_purchase".

Posteriorment, es van afegir les dades amb un "INSERT INTO SELECT". Aquest csv contenia atributs de les entitats compra, article, art\_arena, cofre, bundle i emoticones.

Després es van afegir els atributs de l'entitat article, aquests son la PK, el nom i el preu. Per a l'entitat art\_arena es va afegir la seva PK, i per a que agafes valors diferents vam afegir un "SELECT DISTINCT". A cofre es va afegir el seu nom, el temps, la raresa i la quantitat de cartes, en l'entitat bundle es va importar els atributs or i gemmes. I finalment a l'entitat emoticones es van afegir el nom de la imatge i la seva direcció. A les entitats cofre, arena, emoticones i bundle s'ha afegit un WHERE amb un dels valors que s'afegeix indicant que si es null aquest valor no s'afegeixi i si no ho és s'afegeixi. Això s'ha fet amb la condició també es va afegir un "WHERE valor IS NOT NULL".

### ***Players.csv***

La importació del csv players ha estat resolta de la següent manera:

Primer de tot, s'ha creat una taula temporal anomenada players on hem definit totes les variables igual que al csv: tag (VARCHAR (255)), name (VARCHAR (255)), experience (INTEGER), trophies (INTEGER), cardnumber (VARCHAR (255)) i cardexpiry (DATE).

Tot seguit, s'ha copiat totes les dades del csv a la taula temporal fent ús de COPY i posant el "path" corresponent d'on es troba el fitxer que es busca.

Després s'ha inserit totes les dades en dues taules diferents per com ho teníem fet al nostre model relacional: la primera és la taula "targeta\_credit" on guardem els atributs numero i caducitat, i la segona taula és "jugador" on allà guardem el tag\_jugador, el nom, la experiència que té, els trofeus que té i el numero de targeta coma clau forània.

Per fer-ho hem fet ús de la comanda "INSERT INTO targeta\_credit" i "INSERT INTO jugador" fent els "SELECTS" corresponents de cada atribut FROM players" en aquets cas.

Finalment, fem un DROP TABLE de players per eliminar la taula temporal creada.

### ***Players\_achievements.csv***

En quan al csv de "Players\_achievements", igual que als altres csv primer s'ha mirat a quines taules corresponia i després el tipus que era cada atribut. Posteriorment s'ha creat la taula temporal anomenada "players\_achievements".

Un cop fet això, amb la comanda "COPY", s'ha copiat els elements del csv "players\_achievements" de la carpeta datasates a la taula temporal creada anomenada igual.

Després s'han importat les dades a l'entitat assoliments. S'ha inserit amb "INSERT INTO SELECT" els atributs; títol, recompensa gemmes, descripció i data.



Finalment s'ha auto incrementat la PK amb la comanda "SERIAL" i s'ha fet un drop table de la taula temporal creada (player\_purchase) i un delete table de la nostra taula (assoliments), si existeixen.

### ***Players\_badge.csv***

En el csv "players\_badge", s'ha mirat primer a quina de les nostres taules pertanyia, als atributs d'aquest csv. Posteriorment s'ha escollit el tipus de cada atribut i s'ha creat una taula temporal anomenada "Players\_badge".

Per a copiar els atributs del datasets del csv, s'ha fet un "COPY" i s'han introduït els atributs a la taula temporal.

Despres s'ha afegit a l'entitat insignia els atributs imatge, titol i data.

Finalment s'ha autoincrementat la PK amb la comanda "SERIAL" i s'ha fet un drop table de la taula temporal creada (player\_badge) i un delete table de la nostra taula (insignia) al principi del codi, d'aquesta manera s'eliminaran les dades si anteriorment existeix.

### ***Playerscards.csv***

Al importar playerscards.csv s'ha decidit utilitzar una taula temporal amb un element per cada columna del csv. S'ha trobat que hi havia algunes cartes les quals estaven a playercards.csv però no és trobaven a cards.csv al fer el INSERT INTO a la taula pertany (relaciona carta amb jugador) donava error donat que alguns noms de cartes que s'estaven inserint a la FK de pertany no és trobaven en la taula carta. Per solucionar-ho al moment de seleccionar les dades a importar és va fer un JOIN del nom de la taula temporal amb el nom de la taula carta. D'aquesta forma s'obligava a que els noms que s'afegien a pertany també estiguessin a carta i és descartaven les que no ho estaven.

### ***Players\_clans.csv***

Per poder importar el csv Players\_clans primer vam haver de crear una taula temporal auxiliar anomenada "jugadors\_clans", una cop creada la taula auxiliar, vam copiar tota la informació utilitzant la funció "COPY", d'aquesta manera es va aconseguir tenir una taula temporal d'on obtenir la informació per així fer els "INSERTS INTO" a les taules on faci falta cada informació.

El primer "INSERT INTO" es va fer a la taula rol, i va consistir en normalitzar la informació en relació a cada rol, per fer-ho vam utilitzar la funció "split\_part" per així poder dividir la part del nom del rol amb la seva descripció i finalment fer la inserció d'informació a la taula rol desde la taula auxiliar.

També es va aprofitar la taula auxiliar per omplir la taula de "forma\_part", per fer-ho primer es va copiar la informació als atributs (tag\_jugador, tag\_clan, data) de la taula i després es va haver de fer un "JOIN" per així relacionar els rols amb els seus respectius IDs de manera que es va poder omplir la columna (id\_rol) de la taula "forma\_part".

Finalment, es fa fer un "DROP TABLE IF EXISTS" de la taula auxiliar per així eliminar-la en cas d'existir.

### ***Players\_clans\_donations.csv***

Aquest fitxer csv estava format per 3 columnes corresponents als atributs (player, clan, gold, date), com que la nostre taula de model físic és idèntica a la del fitxer d'importació podem copiar directament les dades a la taula anomenada "dona", per fer-ho vam utilitzar la funció "COPY" a la taula destí anomenada "dona".

Aquesta taula a part de tenir els atributs del fitxer csv també té un atribut anomenat “id\_donació” aquest és de tipus SERIAL i s’incrementa a cada fila que importem de manera que no hi hauria cap problema en cas que un mateix jugador podria donar diferents vegades a diferents clans.

### ***Players\_decks.csv***

La importació del csv players\_deck ha estat resolta de la següent manera:

Primer de tot, s’ha creat una taula temporal anomenada “playersdeck” on hem definit totes les variables igual que al csv: player (VARCHAR (255)), deck (INTEGER), title (VARCHAR (255)), description (TEXT), date (DATE), card (INTEGER), level (INTEGER).

Tot seguit, s’ha copiat totes les dades del csv a la taula temporal “playersdeck” fent ús de COPY i posant el “path” corresponent d’on es troba el fitxer que es busca.

Després s’ha inserit totes les dades en la taula final creada al model físic pila, la qual té els atributs següents: tag\_jugador (forània), ID\_pila (primària), nom, descripció i data de creació.

Per fer-ho hem fet ús de la comanda “INSERT INTO pila” fent els “SELECTS corresponents de cada atribut FROM playersdeck” en aquests cas.

Finalment, fem un DROP TABLE de “playersdeck” per eliminar la taula temporal creada.

### ***Players\_quests.csv***

Per a importar aquest csv, primer s’ha observat quines dades inclou per a saber a quines taules importar-les. Un cop trobades les taules, s’ha creat una taula temporal introduint totes les dades que portava anomenada “player\_quests” . A mesura que hem entrat les dades, s’ha anat decidint el tipus (varchar, float, integer, date...).

Posteriorment, s’han copiat les dades del csv donat de la carpeta datasets, a la taula temporal que s’acabava de crear.

Aquest csv, inclou atributs de les entitats, missio i depen.

Per a introduir els atributs del csv a les entitats comentades anteriorment s’ha utilitzat la consulta “Insert into select”. Primer s’ha afegit la PK a l’entitat missio i els seus atributs, títol, descripció i requeriment i en l’entitat depen s’ha afegit la id\_missio1 i la id\_missio2.

Finalment s’ha afegit al inici de l’script el delete table de la nostra taula i drop table de la taula temporal, per eliminar les taules un cop executades, si existeixen.

### ***Quest\_arenas.csv***

La importació del csv “quests\_arenas” ha estat resolta de la següent manera:

Primer de tot, s’ha creat una taula temporal anomenada “quests\_arenas” on hem definit totes les variables igual que al csv amb els seus respectius tipus: quest\_id (INTEGER), arena\_id (INTEGER), gold (INTEGER), experience (INTEGER).

Tot seguit, s'ha copiat totes les dades del csv a la taula temporal fent ús de COPY i posant el "path" corresponent d'on es troba el fitxer que es busca.

Després s'ha inserit totes les dades en dues taules diferents per com ho teníem fet al nostre model relacional: la primera és la taula "missio" on guardem l'atribut del id\_missio, i la segona taula és "completen" on allà guardem id\_missio, id\_arena, or i experiència.

Per fer-ho hem fet ús de la comanda "INSERT INTO missio" i "INSERT INTO completen" fent els "SELECTS corresponents de cada atribut FROM quests\_arenas" en aquets cas.

Finalment, fem un DROP TABLE de "quests\_arenas" per eliminar la taula temporal creada.

### ***Seasons.csv***

El seasons.csv és un data set amb només 3 columnes simples les qual indiquen el nom de la temporada, quan va començar i quan va acabar. S'han importat aquestes tres columnes directament a les tres columnes de la taula temporada amb la comanda COPY de tal manera que el nom és la primary key de la taula.

### ***Shared\_decks.csv***

La importació d'aquest csv no ha suposat gaire complicació donat a la seva simplicitat. Hi ha dues columnes, una on s'indica el la pila i l'altre on s'indica amb quin jugador s'ha compartit. Donat que el conjunt de les dues columnes formen la clau primària i aquestes son claus foranies primer és necessari que prèviament s'hagin inserit les dades referents als jugadors i a les piles per tal de poder realitzar la relació amb la clau forania. Donat que les columnes del csv eren les mateixes de la taula pila s'ha pogut inserir les dades directament amb el COPY a la taula de pila sense haver d'utilitzar una taula temporal.

### ***Technologies.csv***

Per poder importar el fitxer amb la informació de les tecnologies primer es va haver de crear una taula temporal anomenada "technologies" amb tots els atributs necessaris per importar tota la informació, una vegada amb la informació afegida a la taula auxiliar es va realitzar una copia dels atributs (nom\_millora, descripcio, cost, mod\_damage, mod\_hit\_speed, mod\_radius, mod\_spawn\_damage, mod\_lifetime) a la taula de millores.

Després es va fer una altre copia dels atributs (id\_tecnologia, nivell\_maxim) de la taula auxiliar a la taula tecnologia per així tenir els atributs normalitzats, i finalment es va fer una última copia de la taula temporal a la taula "requereix\_tecnologia" per així guardar-ne els atributs (id\_tecnologia\_nova, id\_tecnologia\_requerida, nivell\_prerequisit).

Un cop copiada tota la informació de la taula auxiliar es fa un DROP TABLE IF EXISTS de la taula "technologies".

## 7 Validació de la base de dades

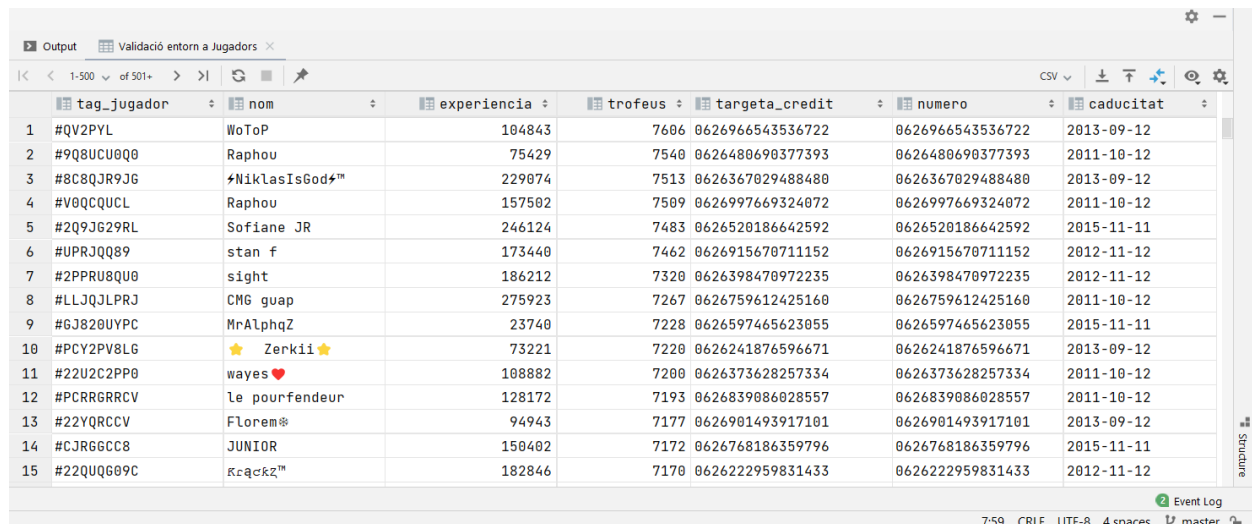
### Jugadors

·JUGADOR:

Primer de tot, hem començat validant el csv de “players”. Per fer-ho, hem mostrat les dades que s’han guardat a la nostres taules “jugador” i “targeta\_credit” ja que en el nostre model relacional ho vam separar en dos entitats. Per fer-ho hem fet servir la següent consulta:

```
SELECT tag_jugador, nom, experiencia, trofeus, targeta_credit, numero, caducitat
FROM jugador as j
JOIN targeta_credit AS tc ON j.targeta_credit = tc.numero;
```

A continuació podem observar que el output que ens retorna la base de dades és amb les mateixes dades que el fitxer “players.csv”.



	tag_jugador	nom	experiencia	trofeus	targeta_credit	numero	caducitat
1	#QV2PYL	WoToP	104843	7606	0626966543536722	0626966543536722	2013-09-12
2	#9Q8UCU0Q0	Raphou	75429	7540	0626480690377393	0626480690377393	2011-10-12
3	#8C8QJR9JG	NiklasIsGod™	229074	7513	0626367029488480	0626367029488480	2013-09-12
4	#V0QCQUCL	Raphou	157502	7509	0626997669324072	0626997669324072	2011-10-12
5	#2Q9JG29RL	Sofiane JR	246124	7483	0626520186642592	0626520186642592	2015-11-11
6	#UPRJQ089	stan f	173440	7462	0626915670711152	0626915670711152	2012-11-12
7	#2PPRU8QU0	sight	186212	7320	0626398470972235	0626398470972235	2012-11-12
8	#LLJQJLPRJ	CMG guap	275923	7267	0626759612425160	0626759612425160	2011-10-12
9	#6J820UYPC	MrAlphqZ	23740	7228	0626597465623055	0626597465623055	2015-11-11
10	#PCY2PV8LG	★ Zerkii ★	73221	7220	0626241876596671	0626241876596671	2013-09-12
11	#22U2C2PP0	wayes ❤	108882	7200	0626373628257334	0626373628257334	2011-10-12
12	#PCRRGRRCV	le pourfendeur	128172	7193	0626839086028557	0626839086028557	2011-10-12
13	#22YQRCCV	Florem®	94943	7177	0626901493917101	0626901493917101	2013-09-12
14	#CJR6GCC8	JUNIOR	150402	7172	0626768186359796	0626768186359796	2015-11-11
15	#22QUQ609C	Ксқсқз™	102846	7170	0626222959831433	0626222959831433	2012-11-12

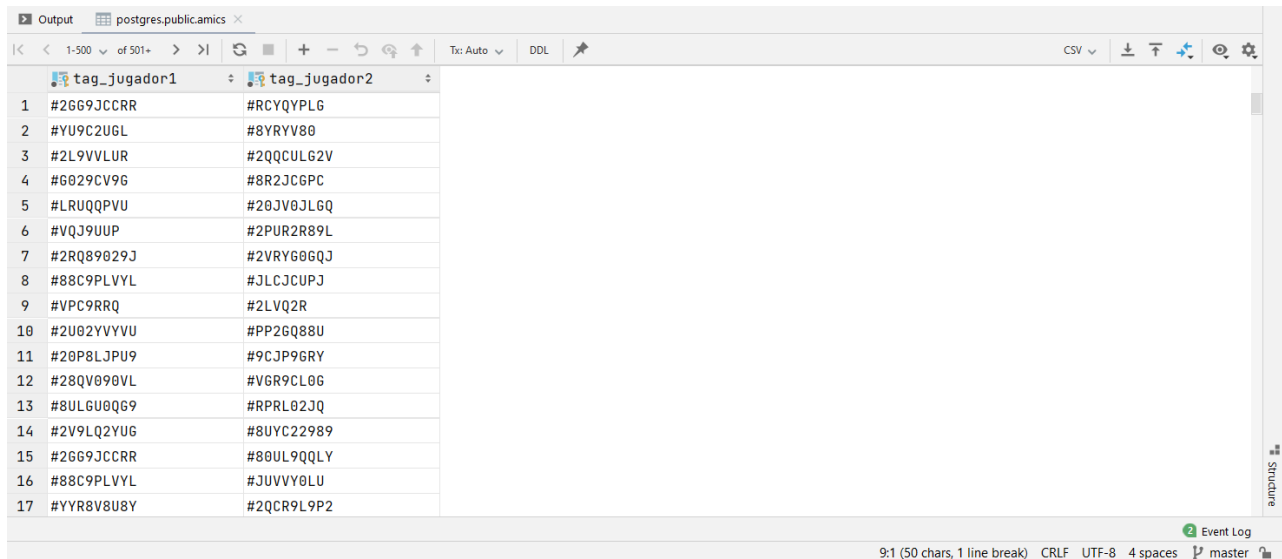
S’ha fet ús d’un JOIN ja que com bé s’ha comentat anteriorment, els atributs que conté el csv de “players”, els teníem separats en dues taules, per tant, per tal de mostrar-les totes juntes en una mateixa taula al demanar les dades, fem ús del JOIN. També recalcar que hem posat un “alias” (AS) per a què sigui més fàcil i ràpid de codificar.

·AMICS:

Després hem fet també una petita i ràpida validació del fitxer “friends.csv” el qual s’han portat les seves dades a la nostra taula “amics”. Aquesta validació ha estat molt ràpida ja que teníem el mateix al model relacional que al fitxer. La consulta ha estat la següent:

```
SELECT tag_jugador1, tag_jugador2
FROM amics;
```

A continuació la captura on es mostra que ens retorna els mateixos valors per als atributs del fitxer:



	tag_jugador1	tag_jugador2
1	#2G69JCCRR	#RCYQYPLG
2	#YU9C2UGL	#8YRYV80
3	#2L9VVLUR	#2QQCULG2V
4	#G029CV96	#8R2JG6PC
5	#LRUQPVVU	#20JV0JLGQ
6	#VQJ9UUP	#2PUR2R89L
7	#2RQ89029J	#2VRYG0GQJ
8	#88C9PLVYL	#JLCJCUPJ
9	#VPC9RRQ	#2LVQ2R
10	#2U02YVYVU	#PP2GQ88U
11	#20P8LJPU9	#9CJP96RY
12	#28QV090VL	#VGR9CL0G
13	#8ULGU0Q69	#RPRL02JQ
14	#2V9LQ2YUG	#8UYC22989
15	#2G69JCCRR	#80UL9QQLY
16	#88C9PLVYL	#JUUVY0LU
17	#YYR8V8U8Y	#2QCR9L9P2

## Cartes

Amb la següent consulta s'obté el nombre de cartes que son de tipus edifici per cada jugador. S'han ajuntat amb JOINS les taules jugador i pertany mitjançant el tag del jugador (identificador únic) i la taula pertany amb la taula edifici mitjançant el nom de la carta (també identificador únic), no ha sigut necessari utilitzar la taula carta donat que al ser una generalització l'identificador és el mateix. Un ajuntades les taules s'ha agrupat per jugador mostrant el seu tag, el seu nom i el recompte d'edificis. Casualment ha sortit que hi ha 13 cartes de tipus edifici en tots els jugadors.

```
SELECT j.tag_jugador, j.nom, COUNT(ed.nom) AS num_c_edifici
FROM jugador AS j
JOIN pertany AS p 1<->1..n: on j.tag_jugador = p.tag_jugador
JOIN edifici AS ed ON ed.nom = p.nom_carta
GROUP BY j.tag_jugador;
```

	tag_jugador	nom	num_c_edifici
1	#PJUJCVR	Extreme	13
2	#99Q8CU	Kaique™	13
3	#8092RGP8J	Lil Kraz	13
4	#828L9RJG	UA I Joker	13
5	#88U0V8C	alex	13
6	#9JCVUYP	Alien✂ Fedez™	13
7	#998LUCJC	ShadowFighter	13

A posteriori s'ha mostrat quantes piles compartides te cada jugador. Per dur-ho a terme ha sigut necessari crear una taula gran ajuntant jugador i pila mitjançant el tag del jugador i comparteixen amb pila amb el id de la pila. Després s'ha tornat a agrupar per jugador mostrant el seu tag i el seu nom de nou i comptant el nombre diferent de piles que tenim. El DISTINCT és necessari donat que un jugador pot compartir una pila amb varis jugadors i sense ell aquestes les comptaríem per doble.

```
SELECT j.tag_jugador, j.nom, COUNT(DISTINCT c.id_pila) AS piles_compartides
FROM jugador AS j
JOIN pila p 1<->1..n: on j.tag_jugador = p.tag_jugador
JOIN comparteixen c 1<->1..n: on p.id_pila = c.id_pila
GROUP BY j.tag_jugador;
```

	tag_ju...	nom	piles...
6	#20L8Y9QY	*WEN*	2
7	#20RJCRL9	GalloMuerto	1
8	#20YJGL9V	zDeadpoolz	2
9	#2220RUV9Y	ViscaBarca ❤	3
10	#222VRRYJ	Gal	1
11	#228J02890	Zekel CR ⭐	1

Una altra consulta realitzada és la obtenció de un jugador en quantes piles diferents hi te la mateixa carta. En aquest cas és necessari ajuntar les taules carta i formen amb el nom de la carta i la taula formen i pila amb l'id de la pila. Per acabar primer s'ha agrupat per cartes i per jugadors i mostrant el tag del jugador, el nom de la carta i en quantes piles es troba.

```
SELECT p.tag_jugador, c.nom AS nom_carta, COUNT(f.nom_carta) AS total
FROM carta AS c
JOIN formen AS f 1<->1..n: on c.nom = f.nom_carta
JOIN pila p 1..n<->1: on f.id_pila = p.id_pila
GROUP BY c.nom, p.tag_jugador;
```

	tag_jugador	nom_carta	total
40	#2RQ89029J	Tesla	1
41	#9PYJVPLV	Graveyard	2
42	#8Y9L200U	Magic Archer	1
43	#Q88YUVLG	Giant	2
44	#PG0QC29YR	Bomber	1

Per últim s'ha comptabilitzat per cada piles quantes rareses de cartes diferents hi ha. Ha sigut necessari ajuntar la taula pila, carta i formen tal i com s'ha fet en la consulta anterior, a continuació s'ha agrupat per piles mostrant el seu nom i comptant un sol cop, amb COUNT(DISTINCT), cada raresa diferent que s'ha trobat a la pila.

```
SELECT p.nom AS pila, COUNT(DISTINCT c.raresa) AS rareses_diferents
FROM pila AS p
JOIN formen f 1<->1..n: on p.id_pila = f.id_pila
JOIN carta c 1..n<->1: on c.nom = f.nom_carta
GROUP BY p.id_pila
ORDER BY rareses_diferents;
```

	pila	rarese...
64	Comunidad Valenciana zombies	2
65	Navarra dwarves	2
66	Principado de Asturias warlocks	2
67	Región de Murcia exorcists	2
68	Baleares owls	3
69	Cataluña zombies	3

Amb les quatre consultes anteriors s'ha pogut mostrar que hi ha una correcte generalització amb article i edificis i mantenen el mateix identificador (donat a l'espai limitat per explicar consultes s'ha obviat els altres elements de la generalització), tant les cartes com les piles estan correctament relacionades amb els jugadors i les piles amb les cartes entre si també donat que les dades obtingudes tenen sentit i corresponen amb les del CSVs.

## Clans

### JUGADORS DE CADA CLAN:

En aquesta primera consulta s'ha volgut comprovar que realment es guarda correctament el jugador de cada clan, de manera que hem relacionat les taules de clan, forma\_part, jugador i dona, així aconseguim veure a quin clan pertany cada jugador, amb el seu “nickname”, el seu id de jugador i la suma de donacions que ha fet.

```
SELECT clan.nom as nom_clan, jugador.tag_jugador, jugador.nom, sum(dona.quantitat) AS
total_donat
FROM clan
JOIN forma_part ON forma_part.tag_clan = clan.tag_clan
JOIN jugador ON jugador.tag_jugador = forma_part.tag_jugador
JOIN dona ON dona.tag_jugador = jugador.tag_jugador
JOIN rol ON rol.id_rol = forma_part.id_rol
GROUP BY jugador.tag_jugador, clan.tag_clan
ORDER BY total_donat DESC
LIMIT 20;
```

	nom_clan	tag_jugador	nom	total_donat
1	Team Legacy	#20L8Y9QY	*WEN*	440
2	Spacestation	#8L08LCP28	Elite Sparky	435
3	Outcasts™	#9ULPU0QL	*LUMINARY*	435
4	Nova l Hispania	#26JC9PR28	CHN   Trex✦	425
5	darkzero	#82LP8CPJ8	Perfect300	420
6	Nova I Aryayi	#6Y2VG2U0	☞MOH@MM@D☞	415
7	Nova l Hispania	#2JRYU6Y9V	CHINO يا رتزي	410
8	NoA	#P8YQU9LL2	Reincarnation	410
9	Undiscovered™	#YQCL8VRL	Jason™	410
10	Team Solid	#28QV090VL	Fredson	410
11	Spacestation	#69GVYLQP	indianboy23	405
12	darkzero	#6QJY00YP	kayden	405
13	Spacestation	#2VLC20U9V	Matthew✦❤️™	405
14	NoA	#8C9RQYV2J	Nera	405
15	AK Syndicate	#20JV0JLGQ	Andrés Fabián ✦	405
16	vikings br	#2CVLVGVRL	Avengers	405
17	NoA	#8982QVPQV	⚡Tadeuz⚡	405
18	QLASH Eclipse	#8VUPVJ2R	生活zz🇸🇦	405
19	NoA	#PV8LRVCJU	ホリグ	405
20	Always Baked	#92VY9UPG	nick	400

Amb el que ens retorna la consulta realitzada podem comprovar que la taula està ben importada ja que es pot veure com cada jugador pertany a un clan en concret junt amb els seu “nickname”. També podem comprovar que la suma total de donacions és correcte

## CLAN TÉ ESTRUCTURA:

En aquesta altre consulta es comprova les estructures que té cada clan junt amb la informació de cada estructura, de manera que relacionem les taules de clan, tenen\_estructura, tecnologia i millora.

```
SELECT clan.nom, estructura.id_estructura, estructura.minim_trofeus, millora.descripcio
FROM clan
JOIN tenen_estructura ON clan.tag_clan = tenen_estructura.tag_clan
JOIN estructura ON tenen_estructura.id_estructura = estructura.id_estructura
JOIN millora ON millora.nom_millora = estructura.id_estructura
ORDER BY clan.tag_clan;
```

L'output que s'obté d'aquesta consulta és el següent:

	nom	id_estructura	minim_trofeus	descripcio
1	Rocket's	Wat	1050	Wat modifies the cards with the following modifiers...
2	Rocket's	Stonehenge	1250	Stonehenge modifies the cards with the following mo...
3	Rocket's	Monument	1200	Monument modifies the cards with the following modi...
4	Rocket's	Mahabodhi Temple	1050	Mahabodhi Temple modifies the cards with the follow...
5	Rocket's	Broadcast Center	1300	Broadcast Center modifies the cards with the follow...
6	Rocket's	Armory	1300	Armory modifies the cards with the following modifi...
7	Rocket's	Ruhr Valley	1050	Ruhr Valley modifies the cards with the following m...
8	Rocket's	Gurdwara	1050	Gurdwara modifies the cards with the following modi...
9	Rocket's	Airport	1250	Airport modifies the cards with the following modif...
10	Rocket's	Alhambra	1150	Alhambra modifies the cards with the following modi...
11	Rocket's	Terracotta Army	1050	Terracotta Army modifies the cards with the followi...
12	Rocket's	Lighthouse	1050	Lighthouse modifies the cards with the following mo...
13	Rocket's	Big Ben	1050	Big Ben modifies the cards with the following modif...
14	Rocket's	Great Zimbabwe	1150	Great Zimbabwe modifies the cards with the followin...
15	Rocket's	Ancient Walls	1150	Ancient Walls modifies the cards with the following...
16	Rocket's	Pyramids	1150	Pyramids modifies the cards with the following modi...
17	Rocket's	Potala Palace	1150	Potala Palace modifies the cards with the following...
18	Rocket's	Madrasa	1250	Madrasa modifies the cards with the following modif...
19	Rocket's	Seaport	1300	Seaport modifies the cards with the following modif...
20	Nova l Hispania	Zoo	1200	Zoo modifies the cards with the following modifiers...
21	Nova l Hispania	Amphitheater	1150	Amphitheater modifies the cards with the following ...
22	Nova l Hispania	Granary	1100	Granary modifies the cards with the following modif...
23	Nova l Hispania	Eiffel Tower	1150	Eiffel Tower modifies the cards with the following ...

Gràcies al resultat de la consulta podem comprovar el funcionament ja que podem veure el nom del clan junt amb les estructures que té, el nombre mínim de trofeus que necessita per obtenir l'estructura i finalment la descripció de cada estructura.



## INFORMACIÓ DE LES TECNOLOGIES:

En aquesta última consulta es comprova el prerequisit de cada tecnologia junt amb el seu nivell prerequisit i el seu cost. A més s'ordena de manera decreixent i és limita el resultat a 20 files.

```
SELECT requereix_tecnologia.id_tecnologia_nova AS nova_millora, millora.cost,  
requereix_tecnologia.nivell_prerequisit, requereix_tecnologia.id_tecnologia_requerida AS  
tecnologia_requerida  
FROM tecnologia  
JOIN requereix_tecnologia ON tecnologia.id_tecnologia =  
requereix_tecnologia.id_tecnologia_nova  
JOIN millora ON millora.nom_millora = tecnologia.id_tecnologia  
ORDER BY millora.cost DESC  
LIMIT 20;
```

	nova_millora	cost	nivell_prerequisit	tecnologia_requerida
1	Future Tech	2500	2	Satellites
2	Nanotechnology	2155	2	Composites
3	Nuclear Fusion	2155	2	Lasers
4	Robotics	2155	5	Computers
5	Stealth Technology	1850	5	Synthetic Materials
6	Telecommunications	1850	5	Computers
7	Composites	1850	5	Synthetic Materials
8	Lasers	1850	5	Nuclear Fission
9	Guidance Systems	1850	3	Rocketry
10	Satellites	1850	3	Rocketry
11	Nuclear Fission	1580	3	Combined Arms
12	Synthetic Materials	1580	3	Plastics
13	Computers	1580	4	Electricity
14	Combined Arms	1410	4	Combustion
15	Advanced Ballistics	1410	6	Replaceable Parts
16	Plastics	1410	4	Combustion
17	Rocketry	1410	4	Radio
18	Advanced Flight	1410	4	Radio
19	Combustion	1250	6	Steel
20	Chemistry	1250	4	Sanitation

Com es pot veure en el resultat de la consulta a la primera columna trobem la nova tecnologia, a la segona columna el seu cost corresponent, a la tercera el nivell prerequisit de la tecnologia requerida i finalment a la última columna es pot veure quina tecnologia es necessita. Finalment es pot comprovar que ho ordena pel cost de la nova millora i limita el resultat a 20 files correctament.

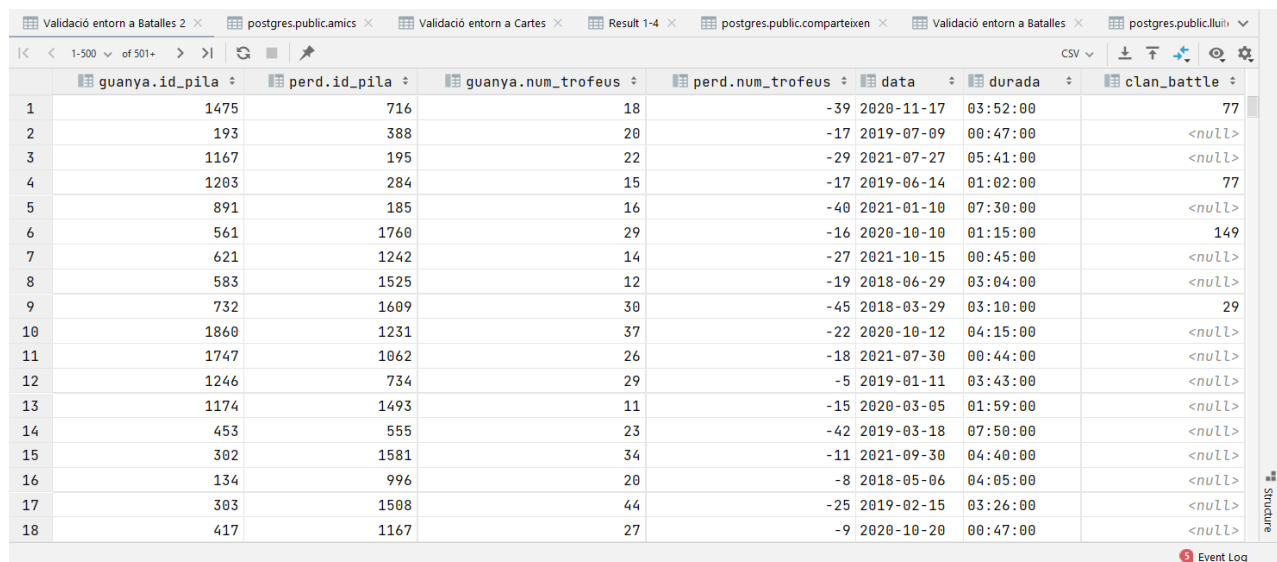
## Batalles

### ·BATALLA:

Primer de tot hem començat amb, com no, el fitxer “battles.csv”, que és el que engloba totes les batalles. Per tal de poder mostrar els atributs que se’ns proporcionaven al fitxer, hem hagut de mostrar els atributs de tres taules diferents creades per nosaltres. Això ha estat degut a que el nostre model relacional estava plantejat d’una manera diferent i per mostrar-ho tal i com es demana, s’ha hagut de realitzar la següent consulta:

```
SELECT guanya.id_pila, perd.id_pila, guanya.num_trofeus, perd.num_trofeus, batalla.data,
batalla.durada, batalla.clan_battles
FROM batalla
JOIN guanya ON batalla.id_batalla = guanya.id_batalla
JOIN perd ON batalla.id_batalla = perd.id_batalla;
```

A continuació podem observar el output que ens retorna amb els valors del fitxer:



	guanya.id_pila	perd.id_pila	guanya.num_trofeus	perd.num_trofeus	data	durada	clan_battle
1	1475	716	18	-39	2020-11-17	03:52:00	77
2	193	388	20	-17	2019-07-09	00:47:00	<null>
3	1167	195	22	-29	2021-07-27	05:41:00	<null>
4	1203	284	15	-17	2019-06-14	01:02:00	77
5	891	185	16	-40	2021-01-10	07:30:00	<null>
6	561	1760	29	-16	2020-10-10	01:15:00	149
7	621	1242	14	-27	2021-10-15	00:45:00	<null>
8	583	1525	12	-19	2018-06-29	03:04:00	<null>
9	732	1609	30	-45	2018-03-29	03:10:00	29
10	1860	1231	37	-22	2020-10-12	04:15:00	<null>
11	1747	1062	26	-18	2021-07-30	00:44:00	<null>
12	1246	734	29	-5	2019-01-11	03:43:00	<null>
13	1174	1493	11	-15	2020-03-05	01:59:00	<null>
14	453	555	23	-42	2019-03-18	07:50:00	<null>
15	302	1581	34	-11	2021-09-30	04:40:00	<null>
16	134	996	20	-8	2018-05-06	04:05:00	<null>
17	303	1508	44	-25	2019-02-15	03:26:00	<null>
18	417	1167	27	-9	2020-10-20	00:47:00	<null>

S’ha fet ús de dos JOIN ja que com bé s’ha comentat anteriorment, els atributs que conté el csv de “battles”, els teníem separats en tres taules, per tant, per tal de mostrar-les totes juntes en una mateixa taula al demanar les dades, fem ús del JOIN. Els hem unit mitjançant els id\_batalla, ja que és un atribut que ambdues taules comparteixen.

### ·BATALLS DE CLAN:

Després també hem fet una petita validació amb el fitxer “clan\_battles.csv” el qual té els mateixos atributs que nosaltres hem declarat per a la taula “lluiten”. Per tant, la comanda a realitzar, ha quedat molt senzilla i és la següent:

```
SELECT tag_clan, id_batalla, data_inici, data_fi
FROM lluiten;
```

A continuació es pot observar com quedaria la nostra taula omplerta després de fer la comanda:

tag_clan	id_batalla	data_inici	data_fi
1 #VQCQJJP	1	2021-10-11	2022-06-05
2 #2CQVVCU	1	2021-10-11	2022-06-05
3 #L0QPV	1	2021-10-11	2022-06-05
4 #9G2PVP0R	1	2021-10-11	2022-06-05
5 #QCJ0J9UP	1	2021-10-11	2022-06-05
6 #P0LLG9RG	1	2021-10-11	2022-06-05
7 #L0YJ02YL	2	2020-01-31	2020-03-05
8 #PJY9PP98	2	2020-01-31	2020-03-05
9 #L0QPV	2	2020-01-31	2020-03-05
10 #QCJ0J9UP	3	2020-10-18	2021-10-04
11 #8YLJ8UL2	3	2020-10-18	2021-10-04
12 #P0LLG9RG	4	2020-12-26	2021-01-10
13 #8VUVGPVC	4	2020-12-26	2021-01-10
14 #PPCLCJ69	4	2020-12-26	2021-01-10
15 #L0QPV	4	2020-12-26	2021-01-10
16 #PJY9PP98	4	2020-12-26	2021-01-10
17 #8YLJ8UL2	4	2020-12-26	2021-01-10

## BATALLA GUANYA INSIGNIA

Per importar les dades d'insignies correctament s'ha fet us del csv "Playerbadge". Aquest fitxer conte els atributs d'insignia i de la nostra taula de guanyen. Per a comprovar que les dades fossin correctes s'ha fet una consulta demanant el nom de les insignies i la ruta de la imatge

```
SELECT DISTINCT i.titol as Nom_Insignies,i.imatge as Imatge_Insignia
FROM batalla as b JOIN guanya as g on b.id_batalla = g.id_batalla
JOIN jugador j on g.tag_jugador = j.tag_jugador
JOIN insignia i on b.data = i.data;
```

	nom_insignies	imatge_insignia
1	Played3Years	img/vitae.png
2	Crl20Wins	img/qui.png
3	Grand12Wins	img/molestiae.png
4	LadderTournamentTop1000_1	img/consectetur.png
5	ClanWarWins	img/doloremque.png
6	Crl20Wins2019	img/eum.png
7	LadderTop1000_3	img/doloribus.png
8	LadderTournamentTop1000_3	img/explicabo.png
9	Played1Year	img/blanditiis.png
10	TopLeague	img/autem.png
11	LadderTop1000_1	img/quia.png

## Arenes

### INFORMACIÓ FILTRADA DE CADA ARENA:

En aquesta primera consulta s'ha comprovat que la informació de cada arena junt amb el seu arena\_pack és correcte, per fer-ho, s'ha relaciona la taula arena amb la taula arena\_pack\_arena, seguidament s'ha fet un GROUP BY amb l'id de l'arena junt amb un SUM amb l'or que es guanya dins de cada arena. D'aquesta manera es pot veure les 15 arenes on es rep més or.

```
SELECT arena.titol AS nom_arena, SUM(arena_pack_arena.or_) AS or_arena_total
FROM arena
JOIN arena_pack_arena ON arena.id_arena = arena_pack_arena.id_arena
GROUP BY arena.id_arena
ORDER BY or_arena_total DESC;
```

LIMIT 15;

	nom_arena	or_arena_total
1	ArenaPvE - Training Camp	33721
2	Arena_L6 - Master III	30726
3	Arena_L2 - Challenger II	28692
4	TrainingCamp - Training Camp	26763
5	Arena_Cake - Legendary Arena	23976
6	Arena9_Season - Jungle Arena	21972
7	Arena3 - Barbarian Bowl	20176
8	Arena_Shipwreck_2 - Shipwreck Island	19301
9	Arena_Clanboat - Legendary Arena	19254
10	Arena_KingOfTheHillTest - King of the Hill Arena Info	18974
11	Arena_Dream - Legendary Arena	18960
12	ArenaClanWars4 - Clan War Arena	17294
13	Arena_Holiday - Legendary Arena	17022
14	Arena_Legendary - Legendary Arena	16923
15	Arena_Forbidden - Legendary Arena	16469

Com es pot veure en el resultat de la consulta l'arena on més or s'aconsegueix és la "ArenaPvE – Training Camp" amb un total d'or de 33721. També es pot comprovar que s'ordena en funció de la quantitat d'or total i es limita el resultat a 15 files correctament.

## ARENA AMB CARTA:

En aquesta última consulta volem mostrar informació de les diferents arenes relacionades les diverses cartes amb la seva respectiva raresa, per fer-ho afegim un JOIN on posem les taules arena, carta i raresa i fem que la condició de ON les relacioni. A més es posa la condició de que per accedir a les arenes hagi de tenir un nombre mínim de trofeus major a 2300, s'ordena de manera ascendent en funció de la mateixa variable anomenada anteriorment i finalment es limita la consulta a 10 files.

```
SELECT arena.titol, arena.nombre_min , carta.nom, carta.dany, carta.velocitat_atac,
raresa.cost_pujar_nivell
FROM arena
JOIN carta ON arena.id_arena = carta.arena
JOIN raresa ON carta.raresa = raresa.nom
WHERE arena.nombre_min > 2000
ORDER BY arena.nombre_min ASC
LIMIT 10;
```

	titol	nombre_min	nom	dany	velocitat_atac	cost_pujar_nivell
1	Arena8 - Frozen Peak	2300	Archers	38	196	2000
2	Arena8 - Frozen Peak	2300	Bats	154	107	2000
3	Arena_L - Hog Mountain	3000	Goblin Gang	202	52	2000
4	ArenaClanWars2 - Clan War Arena	3000	Mega Minion	129	132	5000
5	ArenaClanWars4 - Clan War Arena	3000	Bowler	58	161	10000
6	ArenaClanWars1 - Clan War Arena	3000	Witch	170	218	10000
7	Arena_L - Hog Mountain	3000	Barbarians	213	234	2000
8	ArenaClanWars3 - Clan War Arena	3000	Night Witch	79	145	20000
9	ArenaClanWars4 - Clan War Arena	3000	Hog Rider	244	135	5000
10	ArenaClanWars3 - Clan War Arena	3000	Rascals	221	107	2000

Com es pot comprovar, podem veure a l'esquerre l'arena a la qual pertany una carta, junt amb el nombre mínim de trofeus de l'arena que no és igual ni inferior a 2000, també es mostra el nom de la carta junt amb algunes propietats dins el joc i finalment es veu el cost que té pujar de nivell el qual es calcula en funció de la raresa de la carta.

## Missions

S'ha començat per realitzar una consulta en la qual obtenim el nombre de missions per cada jugador mostrant el seu tag i nom. En aquest cas només son necessàries la de completen per comptar el nombre de missions i la de jugador per obtenir el nom, s'ha ajuntat amb un JOIN i el tag del jugador. Un cop ajuntades les taules s'ha agrupat per jugador donat que volem informació per cada un d'ells, a continuació hem comptat quants ids de missió tenia cada jugador en la taula completen. Per acabar s'ha ordenat descendent el nombre de missions per mostrar primer els jugadors que n'havien fet més.

```
SELECT j.tag_jugador AS tag_j, j.nom AS nom_j, count(c.id_missio) AS num_missions
FROM completen AS c
JOIN jugador AS j 1..n<->1: on c.tag_jugador = j.tag_jugador
GROUP BY j.tag_jugador
ORDER BY num_missions DESC;
```

	tag_j	nom_j	num_missions
1	#228J02890	Zeke1 CR ★	295
2	#PLLC2VG	Maestro Muten	295
3	#2YCJRP9Q0	⚡️⚡️WEEDZ⚡️⚡️	295
4	#L98JYVL	I am Griingo 🌿	236
5	#UQJ9LUP	Itachi uchiha 🍱	236
6	#2LGUPRCJP	DouggYsTyLe	236

Amb la següent consulta s'ha volgut saber el total d'experiència i or obtingut per un jugador en totes les seves missions, és molt similar a l'anterior amb la diferència que s'ha anat sumat tots els valors d'or i d'experiència de cada jugador. S'ha acabat ordenant mostrant el jugador amb més or guanyat primer.

	tag_j	nom_j	or_guanyat	experiencia_guanyada
1	#228J02890	Zeke1 CR ★	37561	43176095
2	#2YCJRP9Q0	⚡️⚡️WEEDZ⚡️⚡️	37503	46370981
3	#PLLC2VG	Maestro Muten	35308	47064343
4	#UQJ9LUP	Itachi uchiha 🍱	33680	35851778
5	#8VUPVJ2R	生活zzz 🍷	31978	37114612
6	#28PQ96J69	NJ   Coopligth	31963	33407995

```
SELECT j.tag_jugador AS tag_j, j.nom AS nom_j, SUM(c.or_) AS or_guanyat, SUM(c.experiencia) AS experiencia_guanyada
FROM completen AS c
JOIN jugador AS j 1..n<->1: on c.tag_jugador = j.tag_jugador
GROUP BY j.tag_jugador
ORDER BY or_guanyat DESC;
```

A continuació s'ha obtingut per cada missió el nombre d'arenas on s'ha completat. Per fer-ho és necessari ajuntar amb un JOIN la taula missió i la completen. S'ha agrupat per missions i a posteriori s'ha comptat en quantes arenes diferents, amb COUNT(DISTINCT), s'ha realitzat. En totes ha sortit 59, això indica que totes les missions s'han realitzat en totes les arenes almenys un cop.

```
SELECT m.id_missio, m.titol AS titol_missio, COUNT(DISTINCT c.id_arena) AS num_arenas
FROM missio AS m
JOIN completen AS c 1<->1..n: on m.id_missio = c.id_missio
GROUP BY m.id_missio;
```

	id_m...	titol_mi...	num_arenas
1	1	Jock	59
2	2	Elmo	59
3	3	Hube	59
4	4	Alexandre	59
5	5	Barnaby	59
6	6	Dode	59

Amb aquesta última consulta s'ha buscat ajuntar els jugadors amb les missions i les arenes, per dur-ho a terme s'ha buscat les 10 combinacions entre arena i jugador en les quals s'ha obtingut més experiència. S'ha començat per ajuntar les taules arena i completen amb el id\_arena i les taules completen i jugador amb el tag\_jugador. A continuació s'ha agrupat pel id de l'arena i pel tag del jugador i s'ha mostrat el tag i nom del jugador, el títol de l'arena i la suma total d'aquells jugadors en l'arena. Finalment s'ha aplicat un LIMIT 10 per tal de només mostrar els 10 primers.

```
SELECT j.tag_jugador, j.nom, a.titol AS titol_arena, SUM(c.experiencia) AS experiencia
FROM arena AS a
JOIN completen AS c 1<->1..n: on a.id_arena = c.id_arena
JOIN jugador AS j 1..n<->1: on c.tag_jugador = j.tag_jugador
GROUP BY a.id_arena, j.tag_jugador
ORDER BY experiencia DESC
LIMIT 10;
```

	tag_jugador	nom	titol_arena	experiencia
1	#2YCJRP9Q0	WEEDZ	Arena_L10 - Ultimate Champion	1238779
2	#2YCJRP9Q0	WEEDZ	Arena8 - Frozen Peak	1181544
3	#PLLC2VG	Maestro Muten	Arena_L1 - Legendary Arena	1158027
4	#2YCJRP9Q0	WEEDZ	Arena5 - Spell Valley	1154990
5	#PLLC2VG	Maestro Muten	ArenaTvE - Training Camp	1147363
6	#2YCJRP9Q0	WEEDZ	Arena_Cake - Legendary Arena	1128022
7	#PLLC2VG	Maestro Muten	Arena_Electric - Electro Valley	1118205
8	#2YCJRP9Q0	WEEDZ	Arena_Executioner - Legendary Arena	1084351
9	#8VUPVJ2R	生活zz	Arena_Heist - Legendary Arena	1077768
10	#2YCJRP9Q0	WEEDZ	Arena_L6 - Master III	1073563

Amb les consultes anteriors s'ha demostrat que els jugadors, les missions i les arenes ha quedat connectats i relacionats correctament poden obtenir i combinar dades de les diferents taules i relacions sense problemes, per tant és dona com a bona la importació d'aquestes dades.

## Assoliments

Per importar les dades d'assoliments s'ha hagut de fer a partir del csv "playersbadge". Aquest engloba els assoliments i la relació entre la arena, un jugador i un assoliment. Per a demostrar que les dades estaven emplenades correctament s'han fet 2 consultes. La primera consulta s'ha creat igual que el csv importat per a demostrar que les dades emplenades eren iguals al csv. I la segona consulta ha sigut d'un jugador en una arena quina recompensa de gemmes guanya, d'aquesta manera hem pogut comprovar correctament el funcionament a l'entorn d'assoliments.

Consulta1:

```
SELECT ac.tag_jugador, a.titol, a.descripcio, ac.id_arena, ac.data, a.recompensa_gemmes  
FROM assoliment as a JOIN aconseguix ac on a.id_assoliment = ac.id_assoliment;
```

A la següent fotografia podem observar els valors de sortida:

	tag_jugador	titol	descripcio	id_arena	data	recompensa_gemmes
1	#QV2PYL	Team Player	Join a Clan	54000024	2020-07-18	48
2	#QV2PYL	Friend in Need	Donate 2500 cards	54000025	2021-11-06	235
3	#QV2PYL	Road to Glory	Reach Arena 6	54000010	2021-11-02	210
4	#QV2PYL	Gatherer	Collect 40 cards	54000011	2020-09-30	163
5	#QV2PYL	TV Royale	Watch a TV Royale Replay	54000043	2020-04-30	100
6	#QV2PYL	Tournament Rewards	Win 20000 cards from tournaments	54000050	2021-08-04	208
7	#QV2PYL	Tournament Host	Create and finish one tournament	54000002	2020-09-26	123
8	#QV2PYL	Tournament Player	Join a tournament	54000010	2020-11-24	254
9	#QV2PYL	Challenge Streak	Get 12 wins in a single Challenge	54000040	2020-05-20	145
10	#QV2PYL	Practice with Friends	Win 10 Friendly Battles	54000021	2020-03-27	238
11	#QV2PYL	Special Challenge	Participate in 5 unique Special Event Challenges	54000055	2020-07-09	42
12	#QV2PYL	Friend in Need II	Donate 25000 cards	54000028	2020-06-30	191

Com els atributs del csv "playersbadge" a nosaltres ens pertanyien a dos taules diferents, aconseguíem i assoliment. Primer s'ha afegit cada atribut amb la taula corresponent i es per això que al mostrar-ho s'ha fet ús del JOIN per poder mostrar les dos taules en 1. Per unir-les s'ha fet mitjançant la id\_assoliment. S'ha comparat el csv amb aquesta consulta i s'ha pogut comprovar que les dades insertades eren les mateixes que les del csv.

Consulta2:

```
SELECT DISTINCT j.tag_jugador as ID_Jugador, j.nom as Nom_Jugador, a.id_arena as ID_Arena,  
a2.recompensa_gemmes as Assoliment_Gemmes  
FROM jugador as j JOIN aconseguix a on j.tag_jugador = a.tag_jugador  
JOIN assoliment a2 on a.id_assoliment = a2.id_assoliment;
```

A continuació es mostra una imatge de la segona consulta:

	id_jugador	nom_jugador	id_arena	assoliment_gemmes
1	#2RUL98URQ	EA Fernando*	54000016	208
2	#98R0LCGG	CHN   Eragon*	54000057	210
3	#RPYLJJ0U	Psychology2027	54000001	145
4	#8PRGRYPCC	Afonsojr	54000045	163
5	#PLGJU82QC	S A N A	54000057	210
6	#80U90RGV	JORGE CEJA	54000053	210
7	#2JRP8UYP	Winter_❄️	54000055	208
8	#PPRPJCYV	⚡ Anto99 ⚡	54000034	238
9	#2VP9LL2JP	jorge545	54000046	191



En aquesta consulta es pot observar com un jugador amb la seva id i nom en una arena determina aconseguir una recompensa en gemmes. Per a realitzar la consulta s'ha usat el JOIN, d'aquesta manera s'han pogut unir les tres taules. S'han comprovat les dades mirant al csv si estaven correctament emplenades.

### ***Botiga – Paquets d'arena, ofertes, cofres i emoticones***

Primer de tot s'han importat les dades del csv “players\_purchase”, aquest engloba tota la botiga amb els seus paquets, arena, ofertes, cofre i emoticones i també inclou les compres. Per a comprovar que les dades estiguessin ben emplenades s'han fet varies consultes. Així s'ha pogut verificar una importació correcta a l'entorn de botiga.

Consultes Compra i articles:

```
SELECT j.tag_jugador as Id_Jugador, j.nom as Nom, COUNT(c.id_article) as Articles
FROM compren AS c JOIN targeta_credit tc on c.num_targeta = tc.numero
JOIN jugador j on c.tag_jugador = j.tag_jugador
JOIN article a on c.id_article = a.id_article
GROUP BY j.tag_jugador,j.nom
ORDER BY articles desc
LIMIT 10;
```

A continuació es mostra una imatge amb la consulta:

	id_jugador	nom	articles
1	#9LUQP62Y	UA I Viole	14
2	#P2CYLRU9	UA I Hyper	14
3	#89UV8ULY	CHN   Rakan*	14
4	#98YP66JJ	Wilson	14
5	#8CUL88Y08	TickleMyTesla🐼	14
6	#9UCCQ2Q9R	BIG Reichert	13
7	#9CJP96RY	<c2>Stella	13
8	#PJUJCVUR	Extreme	13
9	#RPRL02JQ	pedrin	13

Aquesta consulta ens mostra els 10 jugadors que han comprat més articles. Amb aquestes dades i anant comprovant al csv cada dada amb el valor d'articles comprats s'ha pogut comprovar que les dades s'han emplenat correctament. Per a unir les tres taules s'ha usat un JOIN i per a mostrar les dades per als jugadors s'ha usat un GROUP BY, finalment per ordenar-les de més articles comprat a menys un ORDER BY i per limita a 10 grups un LIMIT.

```
SELECT DISTINCT j.tag_jugador as ID_Jugador, j.nom as Nom, t.numero, a.nom as Nom_article,
a.preu as Preu_article, a.quantitat as Quants, c.descompte as Descompte,c.data_ as Data
FROM jugador as j JOIN targeta_credit as t on j.targeta_credit = t.numero
JOIN compren c on j.tag_jugador = c.tag_jugador and t.numero = c.num_targeta
JOIN article a on c.id_article = a.id_article;
```

	id_jugador	nom	numero	nom_article	preu_article	quants	descompte	data
1	#202C2CU0U	Manoel	0626381901632479	Areca Palm	131.47	16	75.46	2020-06-15
2	#202C2CU0U	Manoel	0626381901632479	Common Woolly Sunflower	61.43	1	57.26	2019-08-17
3	#202C2CU0U	Manoel	0626381901632479	Curlyhead Goldenweed	454.72	10	95.97	2019-10-26
4	#202C2CU0U	Manoel	0626381901632479	Goat Willow	285.39	25	49.98	2021-10-13
5	#202C2CU0U	Manoel	0626381901632479	Starved Fleabane	35.92	26	95.7	2021-03-06
6	#202U2J08Q	ClashKiller125	0626362506723707	Boatlily	291	14	6.84	2019-12-03
7	#202U2J08Q	ClashKiller125	0626362506723707	Early Paspalum	25	13	0.64	2019-11-19
8	#202U2J08Q	ClashKiller125	0626362506723707	Goat Willow	285.39	25	53.32	2021-05-06
9	#202U2J08Q	ClashKiller125	0626362506723707	Hornwort	306.93	30	51.39	2020-03-02

En aquesta consulta s'ha pogut valida correctament la inserció de les dades de la taula compren i article comprovant així amb el csv "player\_purchase" el nom, data, targeta de crèdit usada, descompte, preu i quantitat de la compra i article. En la imatge podem contempla com es mostra quin article ha comprat un jugador amb el id determinat, i aquest article quan ha costat, quants n'ha comprat, el descompte que té i la data de la compra. Per unir tots aquets valors s'ha usat un JOIN.

Consulta jugador, compra, cofre:

```
SELECT j.tag_jugador as ID_Jugador,j.nom as NOM, co.nom_cofre as Paquet_Cofre,a.preu as
Preu_article,
co.raresa as Raresa, co.temps as Temps_Desbloqueig, co.quantitat_cartes as Num_Cartes
FROM article as a join cofre co on a.id_article = co.id_cofre
JOIN compren c on a.id_article = c.id_article
JOIN jugador j on c.tag_jugador = j.tag_jugador;
```

	id_jugador	nom	paquet_cofre	preu_article	raresa	temps_desbloqueig	num_cartes
1	#9QULG002	IIAlfaXII	Clan Chest	183.26	Champion	173	156
2	#QQVY6Q0Y	<c5>swenze12	Epic Lightning Chest	455.46	Epic	123	164
3	#299JLV8L	marlon2.0.02	Lightning Chest	217.1	Common	190	40
4	#8982292LY	imad	King's Chest	121.21	Champion	163	238
5	#8R9YJYRYG	Joel Ar	Legendary King's Chest	284.31	Legendary	163	51
6	#2RLGQ6LC	Maja❤ Snipe*	Legendary Chest	465.39	Legendary	149	226
7	#L09YGU9	KING K00PA	Lightning Chest	217.1	Common	190	40
8	#9QC2Y0J8	thinkshero	Epic Lightning Chest	472.15	Epic	123	164
9	#892P2JPGY	UA I Okok Jr	Legendary Chest	465.39	Legendary	149	226
10	#LGC80RQP	saeed	Draft Chest	489.56	Legendary	110	16
11	#2P0PQCVCPC	Alperen	King's Chest	364.92	Champion	163	238
12	#VCLCY0R	checkhost	Royal Wild Chest	247.77	Champion	165	125
13	#PGY2YV26	TiScaffaz0	Draft Chest	489.56	Legendary	110	16
14	#9RLJ2CG8	SonofGod❤	Golden Chest	25	Epic	51	254
15	#26LLJ9QY	kauan	Lightning Chest	217.1	Common	190	40

Per a poder mostrar la compra d'un cofre ,d'un jugador, el contingut del cofre i el preu, s'ha hagut de utilitzar un JOIN per ajuntar les taules compren, jugador, article i cofre.

Consulta paquet ofertes:

```
SELECT j.tag_jugador as ID_Jugador,b.id_bundle as ID_Bundle,b.or_ as Bundle_Or, b.gemmes as
Bundle_Gemmes
FROM bundle AS b JOIN article as a on b.id_bundle = a.id_article
JOIN compren c on a.id_article = c.id_article
JOIN jugador j on c.tag_jugador = j.tag_jugador;
```

	id_jugador	id_bundle	bundle_or	bundle_gemmes
1	#JU0Q99U0	54	15103	440
2	#8GL2CRGLR	9	8288	117
3	#2VV98UYP0	142	32106	27
4	#6LVCVLJP	91	50416	42
5	#8QYCJR99	43	91872	594
6	#R99YPV2Y	125	45300	116
7	#QJU0P2JJJ	94	29714	265
8	#6GL90Q9J	190	35348	27
9	#J999R29J	28	67840	504
10	#90UL09LP	100	85813	458
11	#8Y6R88JJJ	165	49375	480

Podem veure en la imatge com les dades estan ben emplenades ja que la nostra consulta s'ha fet demanant per a cada jugador quin paquet de bundle ha comprat i aquest paquet quina id té i quina quantitat de gemmes i or hi ha. S'ha comprovat amb el csv "players\_purchases" que fos correcte cada dada. Per poder fer aquesta consulta i unir les 4 taules (jugador, compren, article i bundle) s'ha fet ús del JOIN.

Consulta jugador, compra, emoticones:

```
SELECT j.tag_jugador as ID_Jugador,j.nom, a.nom as Nom_Article, e.nom_imatge
as nom_imatge, e.direccio_imatge as Direccio_imatge
FROM compren as c JOIN article a on c.id_article = a.id_article
JOIN jugador j on c.tag_jugador = j.tag_jugador
JOIN emoticones e on a.id_article = e.id_emoticones
```

	id_jugador	nom	nom_article	nom_imatge	direccio_imatge
1	#2QCUGCPQ	Chri 2704	San Nicolas Island Buckwheat	Emote34	sfx/emotes/hr_emote_03_d1.ogg
2	#Q28QYVJQ	Shadow Warrior	Java Aerva	Emote8	sfx/emotes/emotes_goblin_02_01_d1.ogg
3	#820PRY28P	Everson	Viburnum	Emote168	sfx/emotes/emotes_s12_pony_d1.ogg
4	#2LR9YP02G	Art the Legend	Maiden Fern	Emote168	sfx/emotes/emotes_s12_pony_d1.ogg
5	#80V0290C8	왕자 Jean Light*	Spiderwort	Emote185	sfx/emotes/emotes_megaKnight_01_01_d1.ogg
6	#2PVC696Q	Trykenny♥ Tilt	Dimple Lichen	Emote162	sfx/emotes/emotes_feb2020_01_sfx2_d1.ogg
7	#696C6C6G	@우람	Northern Marsh Yellowcress	Emote125	sfx/emotes/emotes_miner_01_02_d1.ogg
8	#RQQY2L28	ZEBARDI	Maiden Fern	Emote168	sfx/emotes/emotes_s12_pony_d1.ogg
9	#CGLRRPYL	<c	Ground Nama	Emote57	sfx/emotes/emotes_crl_01_02_d1.ogg
10	#VQ29UYYY	[CMB] Xing	Hybrid Willow	Emote5	sfx/emotes/emotes_goblin_01_02_d1.ogg
11	#2YC9QC8L9	Flopdonk	Bajada Lupine	Emote52	sfx/emotes/emotes_ewiz_01_01_d1.ogg

En aquesta consulta s'ha demanat per a cada jugador que compri una imatge es mostri la id del jugador, el seu nom, el nom de l'article, el nom de la imatge i la seva direcció. Per a poder unir les taules s'ha fet ús del JOIN.

Consulta jugador, compra, paquet\_arena:

```
SELECT DISTINCT j.tag_jugador as ID_Jugador,j.nom as Nom_Jugador, a.preu as Preu_Article,
ap.id_pack AS Paquet_Arena_Id,apa.or_ as Paquet_Arena_Or
FROM jugador as j JOIN compren c on j.tag_jugador = c.tag_jugador
JOIN article as a on a.id_article = c.id_article
JOIN arena_pack as ap on ap.id_pack = a.id_article
JOIN arena_pack_arena as apa on apa.id_arena_pack = ap.id_pack;
```

	id_jugador	nom_jugador	preu_article	paquet_arena_id	paquet_arena_or
1	#8RRCVCGP	BM Master	321.41	85	4607
2	#YC288PYG	Moreau	58.32	45	1362
3	#CPQPPQ28	U DONT KNOW ME™	5.22	60	8986
4	#8YVGGVC8P	XxPredátoRxX	304.57	5	1190
5	#2YCJRP9Q0	🌿太WEEDZ🌿	304.57	5	3729
6	#8RRCVCGP	BM Master	321.41	85	9536
7	#CL2V9U00	TG I ♠️ Jeroo♠️	164.51	93	6769
8	#Q0L92PQ2	⭐️❤️ Moudji❤️⭐️	58.32	45	1362
9	#PQGV0VCQQ	¥ Dennis ¥	164.51	93	6769
10	#8L000LJ9Y	hudi	16.56	82	5860
11	#8UYL0PPJ	Marmitt	423.52	17	456
12	#2G8Q8RCQ8	UA I Nico	5.22	60	8881

Per aquesta consulta s'ha mostrat la id del jugador i el seu nom per la compra d'un paquet d'arena i s'ha també s'ha mostrat el preu d'aquest paquet, la id del paquet i el contingut del paquet. Per a unir totes les taules s'ha fet ús del JOIN.

## 8 Conclusions

### *Recursos emprats*

<b>Etapa</b>	<b>Marc Valsells</b>	<b>Marc Geremias</b>	<b>Irina Aynés</b>	<b>Albert Tomàs</b>	<b>Total</b>
<b>Actualització dels models Entitat-Relació i Relacional</b>	2h	1h	1h	1h	5h
<b>Selecció del tipus de dades</b>	30 min	30 min	30 min	30 min	2h
<b>Codificació del model físic</b>	1h	2h	1h	1h	5h
<b>Importació de la base de dades</b>	5h	5h	5h	5h	20h
<b>Validació de la base de dades</b>	2h 30 min	2h	2h 30min	1h	8h
<b>Memòria</b>	2h	3h	3h	4h 30 min	12h 30 min
<b>Total:</b>	13h	13h 30 min	13h	13h	52h 30min

Aquesta fase dos ha requerit dedicació per part de tots els integrants i, es nota que hem dedicat més temps que a la fase 1. També perquè aquesta fase ha estat una mica més complexa i la codificació porta temps ja que salten molts errors que s'han de controlar.

A l'actualitzar els models entitat-relació i relacional, no s'ha dedicat un moment en concret per fer-ho, simplement hem anat canviant coses a mesura que anàvem implementant el model físic, per això és una mica complicat donar un número d'hores exacte, però hem intentat aproximar-ho al màxim.

Respecte a la selecció del tipus de dades també és una cosa que hem anat fent a mesura que començàvem a codificar, vèiem que era el que ens importaven dels fitxers .csv i posàvem el tipus de dada que creiem més adient, per tant no ha suposat gaire feina.

En quant a la codificació del model físic, aquí sí que ja comencem a pensar amb més deteniment i comencen a saltar errors de sintaxi i diferents, és per això que en la creació de taules s'ha dedicat el temps especificat en la taula.

La importació definitivament és el que mes temps ha portat a cada integrant del grup. Ens vam repartir els fitxers .csv per implementar-los i vam tenir bastants problemes amb errors, sobretot de claus primàries que no estaven ben definides, atributs que faltaven o que sobraven, cosa que ha fet que haguem anat modificant el model relacional i conceptual a mesura que codificàvem la importació sobretot.

La validació ja no ha estat tant costosa, ens vam reunir i vam repartir-nos dos validacions per a cadascú on després ja hauríem d'aprofundir una mica més i posar més de dos SELECTS per integrar per provar diferents taules, també han saltat alguns errors però ja sabíem controlar-los millor degut a que molts eren semblants als que ens sortien a la importació.

Finalment, la memòria, simplement ha estat anar fent poc a poc. Cadascú tenia la seva part per omplir i a mesura que anàvem codificant i millorant els errors, anàvem omplint la memòria per treure feina de sobre.

## ***Lliçons apreses i conclusions***

Creiem que aquesta fase ha estat molt útil per aprendre com funciona una Base de Dades real, fins ara, havíem vist la teoria i a la fase 1, vam posar a prova els nostres coneixements sobre models conceptuals i relacionals, com implementar una idea per fer una base de dades funcional, però en aquesta fase 2, hem après a portar aquesta idea o concepte a la realitat.

El fet d'haver-nos familiaritzat en l'entorn PostgreSQL, ens ha permès portar una base de dades codificada i funcional, així com aprendre com funciona aquest llenguatge de programació i les seves característiques més importants. Al no haver utilitzat mai un programa d'aquest tipus, hem hagut d'acostumar-nos al principi, però a mesura que ho anàvem utilitzant cada cop més, anàvem trobant més fàcil la manera d'utilitzar-lo i les comandes més ràpides per fer el nostre treball el més eficient possible.

Hem après a crear taules amb els seus respectius atributs i els tipus d'aquests (model físic), a importar dades i a validar que aquestes dades estiguin bé mitjançant comandes que hem anat practicant al llarg del semestre amb els exercicis setmanals.

Una altre cosa important és que hem après a utilitzar i treballar amb fitxers .csv, el seu format, com està guardada la informació en ells, com crear-los, com importar-los i copiar les seves dades a una taula, etc... Això ens obre un gran ventall ja que en el món professional estem segurs que es treballa amb fitxers de tot tipus i tenir aquesta petita base de com moure's amb csv és molt important.

En definitiva, l'equip ha treballat molt i estem molt contents del rendiment que cada membre ha donat, ens hem organitzat bastant bé i durant el nadal vam anar reunint-nos per avançar feina i proposar-nos feina per portar-la feta pel següent dia. Esperem que segueixi així durant les pròximes fases 3 i 4 del segon semestre i poder acabar fent una pràctica bona i treballada però sobretot, aprendre el màxim possible que al final és l'objectiu principal de l'assignatura.