CERGY PARIS UNIVERSITÉ

Rapport de projet réseau et base de données

FIRE, THRONE, SORCERY AND STARS project

Par : MANSOUR Amine, SAGOT Thomas et ZOUAOUI Mahdi

L3 informatique CY Cergy Paris université 2021/2022

fait :le 01/12/2021

Table des matières

Contexte du projet :

On se place dans le contexte d'un jeu vidéo en ligne nommé "FIRE, THRONE, SORCERY AND STARS" dont le but est de devenir le plus fort possible en tuant des ennemis pour gagner des équipements pour améliorer son personnage, ou pouvant être vendu avec d'autres joueurs. Ici, on ne s'occupera pas de coder le jeu (cela serait trop long, et ce n'est pas le but du projet), mais on supposera que le jeu existe. Puisque l'on ne créera pas le jeu, on générera les différents éléments de manière aléatoire (notamment les équipements). Chaque joueur est inscrit au jeu avec un compte. Pour accéder à son compte, un joueur doit se connecter avec un log-in et un mot de passe (le mot de passe n'est évidement pas stocké en clair dans la base de donnée). Chaque compte peut avoir plusieurs personnages, qui ont chacun un nom et des caractéristiques principales telles que l'expérience accumulé, la vie du personnage... Le jeu propose un système de guilde : il s'agit d'un rassemblement de joueurs ayant pour but de s'entraider, avec un chef. Les personnages sont classés selon l'expérience (l'expérience est obtenue en jouant) totale acquisse au cours d'une saison (deux personnages avec la même expérience seront considérés ex æquo) d'une durée fixe. Chaque personnage possède des objets (équipés ou non) qu'il pourra mettre en vente selon un certain prix. Dans le jeu, des équipements sont disponibles. Il y a différents types d'équipement notamment arme et armure. Chaque équipement a une apparence, des statiques principales tel que la vitesse d'attaque, et des bonus de divers nature (tel que résistance élémentaire, plus de vie maximale...) qui sont limité par le niveau de l'objet. Le jeu propose un service web permettant de réaliser plusieurs actions. Entre nous avions prévus qu'un joueur, une fois connecté à son compte pourra :

1. Consulter ses différents personnages, ainsi que leur équipement.
2. Récupérer les informations de son compte.
3. Chercher des joueurs selon différent critères (nom de compte, position sur le classement, appartenance à une guilde...)(abandonné).
4. Consulter différentes guildes, pouvoir candidater à certaines guildes (la candidature à été abandonnée, la base de donnée ne permettant que de rejoindre directement (et non demander à rejoindre)).
5. Consulter le classement de différentes saisons.
6. Acheter des équipements d'autre joueurs, et pouvoir effectuer une recherche selon les attributs de l'équipement voulu (prix, nombre de mod...).
7. Décider de mettre un de ces objets (non équipés) en vente, selon un certain prix, ainsi que de changer le prix, ou retirer un objet de la vente (abandonné).
8. Créer un compte.

D'autre part, le client réseau, uniquement accessible à l'administrateur devait à l'origine permettre d'effectuer des recherches avec moins de limites de la base de données (on aurait pu par exemple ici accéder à l'adresse mail de n'importe quel joueur, pour par exemple envoyer un mail à tout les joueurs pour les informer d'une mise à jour quelconque). De plus il aurait été aussi possible de modifier certains éléments de la base, par exemple en modifiant la valeur de certain modificateur pour équilibrer le jeu, ainsi que de bannir un utilisateur, où une guilde. On pourra aussi donner la possibilité de créer des objets pour par exemple récompenser un joueur pour son classement.

Cependant, ceci aurait demandé trop de temps, et on se contentera d'un outils qui permet aux développeurs de bannir des comptes (dans notre contexte, cela revient à effacer un compte et tout ce qu'il possède de la base de données). Enfin, puisqu'il s'agit d'un outil interne à une entreprise, on ne s'occupera de développer une interface graphique en dernier, si le temps le permet, ce qui n'a pas été le cas.

1 Partie Base de Données:

A Architecture du projet :

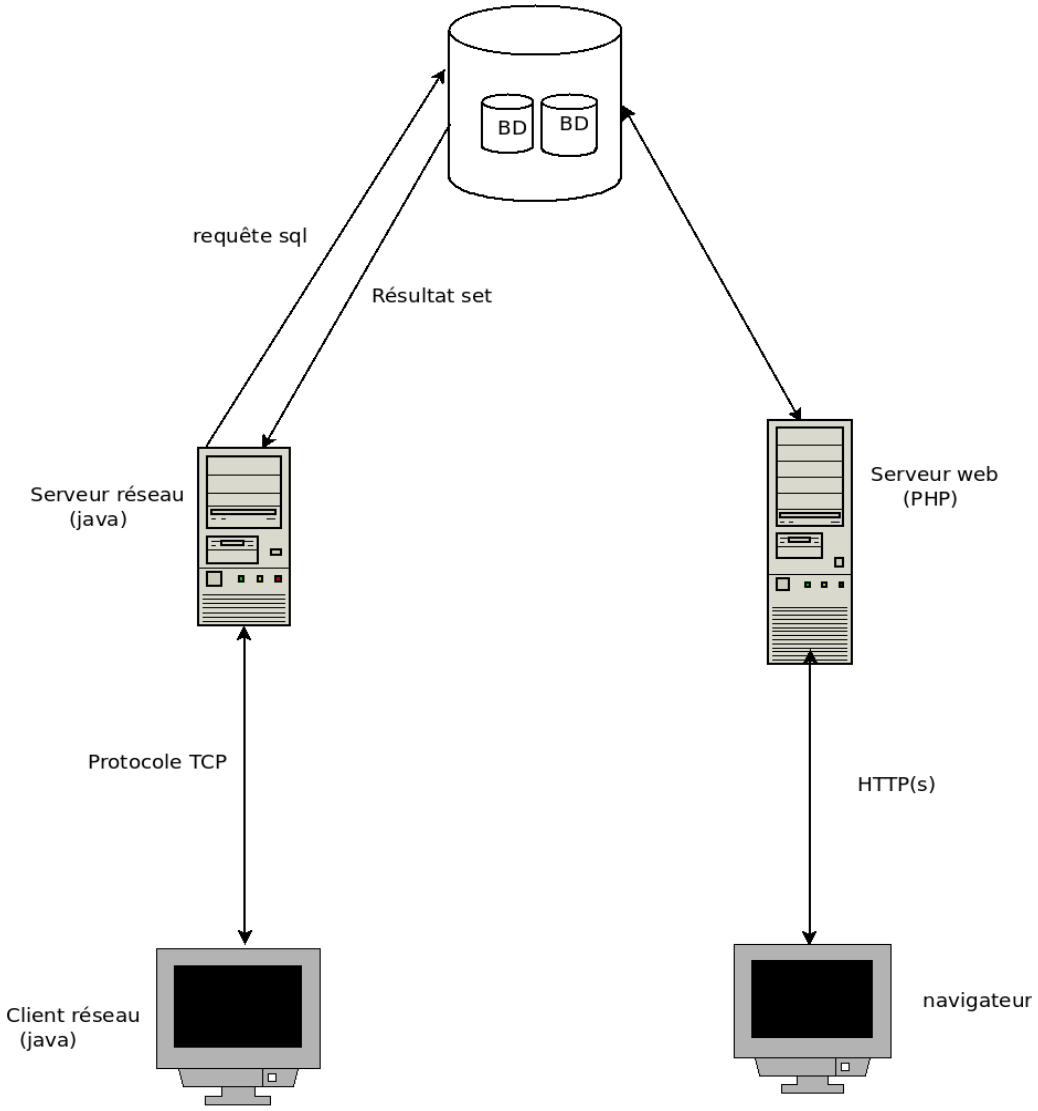

Figure 1: Schéma d'architecture du projet

B Contenu de la base de donnée :

I Dictionnaire de données :

Table / Attribut	Description	type	les remarques
Table : guilde			
nom	Chaîne de caractère correspondant au nom de guilde	VARCHAR(30)	
embleme	Embléme de la guilde (image)	BYTEA	
dateCreationGuilde	Date de la création de la guilde	DATE	
chef	Id du compte du chef de la guilde	CHAR(10)	
Table : Compte			
idCompte	Valeur numérique identifiant de manière unique les comptes de la base de données	CHAR(10)	
nomC	Chaîne de caractère correspondant au nom de compte	VARCHAR(30)	
dateCreation	Date de création du compte	DATE	Après la création du jeu (1/1/2020)
serveur	Région dans laquelle est basée le serveur sur lequel le joueur joue	VARCHAR(10)	
recherchePrec	Enregitrement de la dernière recherche d'objet à acheter via une stucture XML	XML	
eMail	Adresse email de l'utilisateur	VARCHAR(50)	Doit être une adresse valide
mdp	mot de passe de l'utilisateur (après hashage)	CHAR(128)	
solde	Monnaie virtuelle dont dispose un joueur	INT	Suppérieur ou égal à 0
Table:Equipement			
idEquipement	leur numérique identifiant de manière unique les équipements de la base de donné	INT	
nomC	Chaîné de caractère correspondant correspondant au nom des équipements	VARCAHAR(50)	
niveau	niveau d'équipement(plus le niveau est élevé, plus l'équipement est puissant)	INT	Entre 1 et 100
apparence	l'apparence de l'équipement	BYTEA	
table : Arme			
idArme	Valeur numérique identifiant de manière unique les armes	INT	
attaque	Puissance d'Attaque de l'arme	INT	
critChance	Probabilité de faire un coup critique, multipliant les dégats par degatCrit	FLOAT	Comprise entre 0 et 1
degaCrit	Les coups critiques ont leurs dégats multiplié par degatCrit	FLOAT	Supérieure à 1
vitesseAttaque	nombre d'attaque par seconde	FLOAT	Strictement supérieure à 0
table : Modificateur			
tier	Plus le tier d'un modificateur est élevé plus il est puissant	INT	Suppérieur à 1
niveauMod	le niveau minimal de l'objet pour qu'il possède ce modificateur	INT	Suppérieur à 1
type	désigne ce que fait le modificateur(+de vie, plus d'attque...)	VRACHAR(50)	
valeur	La valeur du bonus donné par le modificateur	VARCHAR(30)	
table : armure			
idArmure	Valeur numérique identifiant de manière unique les armurs	INT	
defence	Valeur de la défence de l'armure	INT	
table : Personnage			
idPer	valeur numérique identifiant de manière unique les personnage	INT	
nomPer	Chaîne de caractère correspondant au nom de personnage	VARCHAR(30)	
niveau	niveau de personnage dans le jeu	INT	Compris entre 1 et 100
experience	expérience de personnage dans le jeu	BIGINT	
vieDebase	valeur correspondant a combien de vis possède une pressonage	INT	Suppérieur à 0
table : classement			
position	Place du personnage durant cette saison dans le jeu (classé selon l'expérience	INT	plusieurs presonnages peuvent être classé au même rang
saison	Saison de jeu	INT	
dateDebutS	Début de la saison	DATE	

Figure 2: Dictionnaire de données du projet

II Modèle E/A :

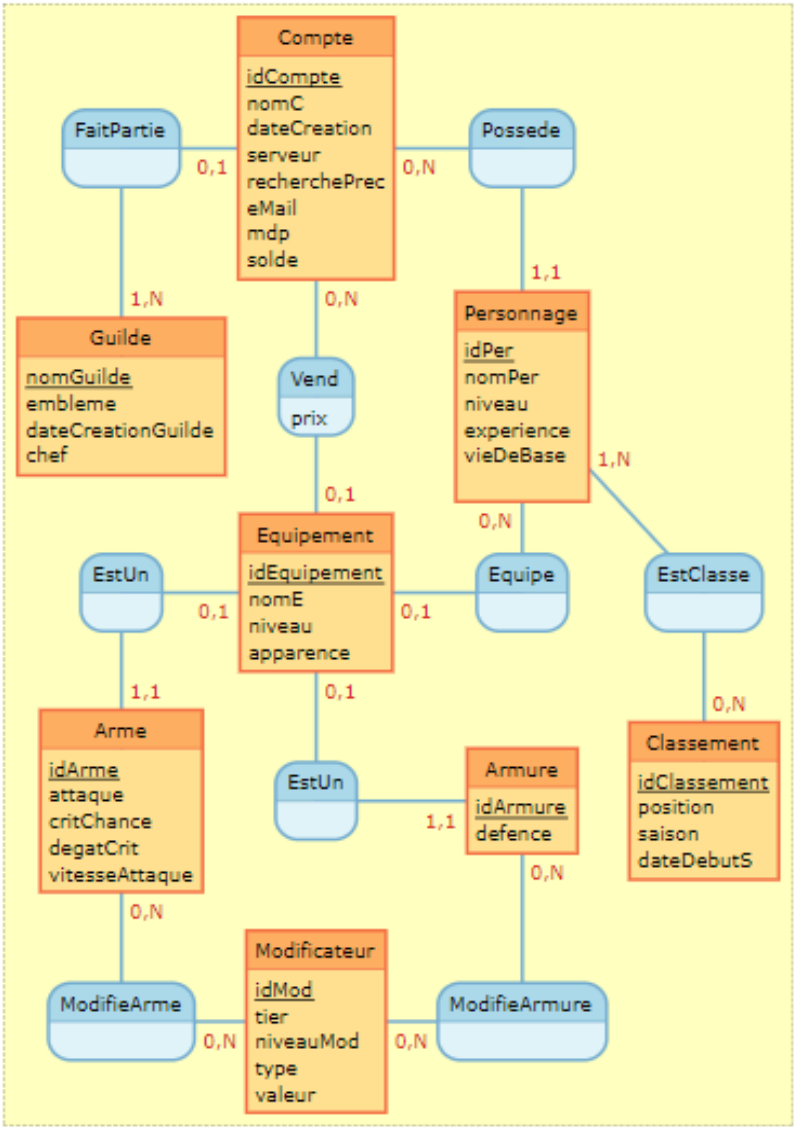

Figure 3: Modélisation E/A du projet

III Schéma relationnel:

- Compte (idCompte, nomC, dateCreation, serveur, rechercheePrec, eMail, mdp, solde, #nom Guilde)
- Guilde (nomGuilde, embleme, dateCreationGuilde, chef)
- Personnage (idPer, nomPer, niveau, experience, vieDeBase, , #idCompte)
- Equipement (idEquipement, nomE, niveau, apparence, #idCompte, prix, #idPer)
- EstClasse (#idPer,# idClassement)
- Arme (# idArme, attaque, critChance, degatCrit, vitesseAttaque, idEquipement)
- Armure (#idArmure, defence, idEquipement)
- Classement (idClassement, position, saison, dateDebutS)
- ModifieArme (#idArme, #idMod)
- Modificateur (idMod, tier, niveauMod, type, valeur)
- ModifieArmure (#idArmure, #idMod)

IV Exemple de jeu de données :

idarme	attaque	critchance	degatcrit	vitesseattaque
1	240	0.1	150	1.4300001
6	410	0.1	150	1.77
11	410	0.1	150	1.77
12	410	0.1	150	1.77
13	405	0.1	150	1.76
14	500	0.1	150	1.95
19	510	0.1	150	1.97
20	505	0.1	150	1.96
21	80	0.1	150	1.11
26	305	0.1	150	1.56

Figure 4: Exemple de jeu de données de la table Arme

idarmure	defence
2	490
3	540
4	500
5	520
7	850
8	850

Figure 5: Exemple de jeu de données de la table Armure

idcompte	nomc	datecreation	serveur	rechercheprec	email	mdp	solde	nomguilde
1	GilletBrun	2019-05-25	NA	NULL	GilletBrun1@email.com	98435006b8819c96199c7be6202e23e9f1e79292771882653...	100	XTJITZOCVBDP
3	LebrunRoux	2019-08-25	NA	NULL	LebrunRoux3@email.com	d2b58f4c80866ec8f5dbe2bdd3982b24067a41cd19d6a7597...	100	TEESHMVLJPUUT
4	HoarauMarty	2020-03-27	EUR	NULL	HoarauMarty4@email.com	c05cc2b151035aff9f09076b7be20e291f5f82467e0ea8201...	100	XTJITZOCVBDP
6	BouvierLeclerc	2020-05-07	EUR	NULL	BouvierLeclerc6@email.com	7ce9cc0d776ab64e9a1d70c1c2797b470c12a0a64839d3dfe...	100	XTJITZOCVBDP
9	MercierNoel	2020-07-14	EUR	NULL	MercierNoel9@email.com	922007261d94ed63e7e42cd8255c7f3a1e8ff828b3c25d402...	100	WVBYXQMPQK
10	ColletRodriguez	2020-04-15	EUR	NULL	ColletRodriguez10@email.com	a62a66b6d1bc88f4573d79e1f21bd2a6f777e826dbd181739...	100	WVBYXQMPQK
13	ThomasRoyer	2020-11-20	NA	NULL	ThomasRoyer13@email.com	6a9473f6858b856554ff1b60ecd1cd02499f7f18790df874e...	100	XTJITZOCVBDP

Figure 6: Exemple de jeu de données de la table Compte

idper	idclassement
23	0
36	1
40	2
56	3
28	4

Figure 7: Exemple de jeu de données de la table EstClasse

nomguilde	embleme	datecreationguilde	chef
WVBYXQMPQK		2020-11-06	26
XTJITZOCVBDP		2020-08-27	11
TEESHMVLJPUUT		2020-11-17	2

Figure 8: Exemple de jeu de données de la table Guilde

idarme	idmod
17	augmentation de la vie,14
17	résistance au chaos,1
17	régénération de vie par seconde,3
23	pourcentage de vitesse d'attaque,2
23	résistance à l'éléctricité,6

Figure 10: Exemple de jeu de données de la table ModifieArme

idarmure	idmod
7	pourcentage de vitesse de déplacement,1
7	augmentation de la vie,15
7	régénération de vie par seconde,1
8	pourcentage de vitesse de déplacement,1
10	augmentation du mana,6

Figure 9: Exemple de jeu de données de la table ModifieArmure

idmod	tier	niveaumod	type	valeur
résistance à la glace,9	9	81	résistance à la glace	29
résistance à la glace,10	10	91	résistance à la glace	32
résistance au chaos,1	1	20	résistance au chaos	5
résistance au chaos,2	2	30	résistance au chaos	8
résistance au chaos,3	3	40	résistance au chaos	11
résistance au chaos,4	4	50	résistance au chaos	14
résistance au chaos,5	5	60	résistance au chaos	17
résistance au chaos,6	6	70	résistance au chaos	20
résistance au chaos,7	7	80	résistance au chaos	23
résistance au chaos,8	8	90	résistance au chaos	26
augmentation du mana,1	1	1	augmentation du mana	10
augmentation du mana,2	2	13	augmentation du mana	15

Figure 11: Exemple de jeu de données de la table Modificateur

25	26	1	2020-01-01
26	27	1	2020-01-01
27	28	1	2020-01-01
28	1	2	2020-01-09
29	2	2	2020-01-09

Figure 13: Exemple de jeu de données de la table classment

idper	nomper	niveau	experience	viedebase	idcompte
1	NCKDBOVMFAGS	46	611473057	2355	1
2	EWUVRZTEJZD	77	24576760848749	6292	3
3	KIFCSKMYRHDFZU	98	26987731728815644	10051	4
4	JDLDADV	13	9767	276	5
5	NASLITTPBQ	58	41244300831	3651	6

Figure 12: Exemple de jeu de données de la table Personnage

V Le script SQL de création de la base de données (*DDL*):

```
CREATE TABLE Guilde(
    nomGuilde VARCHAR(30) NOT NULL,
    embleme BYTEA ,
    dateCreationGuilde DATE,
    chef CHAR(10),
    CONSTRAINT Guilde_pk PRIMARY KEY (nomGuilde)
);
CREATE TABLE Compte(
    idCompte CHAR(10) NOT NULL,
    nomC VARCHAR(30),
    dateCreation DATE,
    serveur VARCHAR(10),
    recherchePrec XML,
    eMail VARCHAR(50),
    mdp CHAR(128),
    solde INT check(solde > 0) ,
    nomGuilde VARCHAR(30),
    CONSTRAINT Compte_pk PRIMARY KEY (idCompte),
    CONSTRAINT  Compte_fk FOREIGN KEY (nomGuilde) REFERENCES Guilde (nomGuilde)
);
CREATE TABLE Personnage (
    idPer INT,
    nomPer VARCHAR(30),
    niveau INT  check(niveau BETWEEN 1 and 100),
    experience bigint,
    vieDebase INT,
    idCompte CHAR(10) NOT NULL,
    CONSTRAINT Personnage_pk PRIMARY KEY (idPer),
    CONSTRAINT Personnage_fk FOREIGN KEY (idCompte) REFERENCES Compte (idCompte)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);
CREATE TABLE Classement (
    idClassement INT,
    position INT  ,
    saison INT    ,
    dateDebutS DATE,
    CONSTRAINT Classement_fk PRIMARY KEY (idClassement)
    );
```

Figure 14: Figure : Le script SQL de création de la base partie 1

```sql
CREATE TABLE Equipement(
    idEquipement INT NOT NULL,
    nomC VARCHAR(50),
    niveau INT check(niveau BETWEEN 1 and 100) ,
    apparence BYTEA ,
    idCompte CHAR(10) NOT NULL,
    prix INT ,
    idPer INT,
    CONSTRAINT Equipement_pk PRIMARY KEY (idEquipement),
    CONSTRAINT Equipement_id_fk FOREIGN KEY (idCompte) REFERENCES  Compte(idCompte)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    CONSTRAINT Equipement_idPer_fk FOREIGN KEY (idPer) REFERENCES  Personnage(idPer)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);
CREATE TABLE EstClasse(
    idPer INT,
    idClassement INT,
    CONSTRAINT EstClasse_pk FOREIGN KEY (idPer) REFERENCES Personnage(idPer)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    CONSTRAINT EstClasse_position_fk FOREIGN KEY (idClassement) REFERENCES Classement(idClassement)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    PRIMARY KEY(idPer, idClassement)
    );
CREATE TABLE Arme (
    idArme INT,
    attaque INT ,
    critChance FLOAT check(critChance >0) ,
    degatCrit FLOAT check(degatCrit > 1),
    vitesseAttaque FLOAT check(vitesseAttaque > 0),
    CONSTRAINT Arme_pk PRIMARY KEY (idArme),
    CONSTRAINT Arme_idArme_fk FOREIGN KEY (idArme) REFERENCES Equipement(idEquipement)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);
CREATE TABLE Modificateur(
    idMod VARCHAR(55),
    tier INT check(tier > 0),
    niveauMod INT,
    type VARCHAR(50),
    valeur VARCHAR(30),
    CONSTRAINT Modificateur_pk PRIMARY KEY (idMod)
);
```

Figure 15: Le script SQL de création de la base partie 2

```sql
CREATE TABLE Armure (
    idArmure INT,
    defence INT ,
    CONSTRAINT Armure_pk PRIMARY KEY (idArmure),
    CONSTRAINT Armure_idArmure_fk FOREIGN KEY (idArmure) REFERENCES  Equipement(idEquipement)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);

CREATE TABLE ModifiArme(
    idArme INT ,
    idMod VARCHAR(55),
    CONSTRAINT ModifiArme_idArme_fk FOREIGN KEY (idArme) REFERENCES Arme(idArme)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    CONSTRAINT ModifiArme_modificateur_fk FOREIGN KEY (idMod) REFERENCES Modificateur (idMod)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    PRIMARY KEY (idArme,idMod)
);

CREATE TABLE ModifiArmure(
    idArmure INT,
    idMod VARCHAR(55),
    CONSTRAINT ModifiArmure_idArmure_fk FOREIGN KEY (idArmure) REFERENCES Equipement (idEquipement)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    CONSTRAINT ModifiArmure_modificateur_fk FOREIGN KEY (idMod) REFERENCES Modificateur (idMod)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
    PRIMARY KEY (idArmure,idMod)
);
```

Figure 16: Le script SQL de création de la base partie 3

VI Les INSERTS SQL :

Étant donné que nous avons un script DML de plus de 1400 lignes nous avons gardé quelques lignes afin d'illustrer notre application du dml .Vous trouverez ci dessous quelques illustration des requêtes sql que nous avons utilisé afin créer notre base de donné , accompagne des capture d'écran des résultat des requêtes au niveau de pgAdmin 4 .

```
INSERT INTO compte Values ('1','LefévreGrondin','19-11-2020','EUR','NULL','LefévreGrondin1@email.com',
    '8fe2b7d1091850c3460901e23f8d2fdcb3793e73e129b01ae3e53657d917d2111840e505c96b0b3100bb3af9a5f8a0bca4102546716594ca865a92288afffb6c','100');
```

Figure 17: requêtes sql permettant l'insertion des données dans la table compte

	idcompte [PK] charact	nomc character varying (3	datecreation date	serveur character	rechercheprec xml	email character varying (50	mdp character (128)	solde integer	nomguilde character varying (30)
1	1	EtienneBlanchard	2020-11-19	EUR	NULL	EtienneBlanchard...	158205468896bc3b...	100	UWMOIKAKFRORSOUZZ

Figure 18: illustration d'insertion des données dans la table compte

s

```
INSERT INTO personnage Values ('1','YVMFMAQFOYBJWVQ','38','35959818','1651','1');
```

Figure 19: requêtes sql permettant l'insertion de données dans la table personnage

	idper [PK] integer	nomper character varying (30)	niveau integer	experience bigint	viedebase integer	idcompte character (10)
1	1	YVMFMAQFOYBJWVQ	38	35959818	1651	1

Figure 20: illustration d'insertion des données dans la table personnage

```
INSERT INTO Guilde Values ('FQRQNVIBRAOSQQ','','18-10-2020','30');
```

Figure 21: requête sql permettant l insertion des données dans la table Guilde

	nomguilde [PK] character varying (30)	embleme bytea	datecreationguilde date	chef character (10)
1	FQRQNVIBRAOSQQ	[binary data]	2020-10-18	30

Figure 22: illustration d'insertion des données dans la table Guilde

```
INSERT INTO Classement Values ('0','1','1','1-1-2020');
```

Figure 23: requêtes sql permettant l'insertion des données dans la table Classement

	idclassement [PK] integer	position integer	saison integer	datedebuts date
1	0	1	1	2020-01-01

Figure 24: illustration d'insertion des données dans la table Classement

```
INSERT INTO EstClasse Values ('1','107');
```

Figure 26: requêtes sql permettant l'insertion des données dans la table EstClasse

	idper [PK] integer	idclassement [PK] integer
1	1	107

Figure 25: illustration d'insertion des donnes dans la table EstClasse

```
INSERT INTO modificateur (idMod, type, tier, valeur, niveauMod)
VALUES ('pourcentage de dégats critiques,1','pourcentage de dégats critiques','1','25','20');
```

Figure 27: requêtes sql permettant l'insertion de données dans la table modificateur

	idmod [PK] character varying (55)	tier integer	niveaumod integer	type character varying (50)	valeur character varying (30)
1	augmentation de la d,fence,1	1	1	augmentation de la d,fence	10

Figure 28: illustration d'insertion des données dans la table modificateur

```
INSERT INTO Equipement Values ('1','longue épée de marche agosiante','38','','1','-1','1');
```

Figure 29:requêtes sql permettant l'insertion de données dans la table Équipement

	idequipement [PK] integer	nomc character varying (50)	niveau integer	apparence bytea	idcompte character (10)	prix integer	idper integer
1	1	longue ,p,e de marche agosiante	38	[binary data]	1	-1	1

Figure 30: illustration d'insertion des données dans la table Équipement

```
INSERT INTO Arme Values ('1','215','0.1','150','1.38');
```

Figure 31: requêtes sql permettant l'insertion de données dans la table Arme

	idarme [PK] integer	attaque integer	critchance double precision	degatcrit double precision	vitesseattaque double precision
1	1	215	0.1	150	1.38

Figure 32: illustration d'insertion des données dans la table Arme

```
INSERT INTO Armure Values ('2','430');
```

Figure 34: requêtes sql permettant l'insertion des données dans la table Armure

	idarmure [PK] integer	defence integer
1	2	430

Figure 33: illustration d'insertion des données dans la table Armure

```
INSERT INTO ModifiArmure Values ('2','pourcentage de dégats critiques,1');
```

Figure 35: requêtes sql permettant l'insertion des données dans la table ModifiArmure

	idarmure [PK] integer	idmod [PK] character varying (55)
1	2	pourcentage de chance de bruler,1

Figure 36: illustration d'insertion des données dans la table ModifiArmure

```
INSERT INTO ModifiArme Values ('11','pourcentage de dégats critiques,1');
```

Figure 37: requêtes sql permettant l'insertion des données dans la table ModifiArme

	idarme [PK] integer	idmod [PK] character varying (55)
1	11	pourcentage de d,gats critiques,1

Figure 38: illustration d'insertion des données dans la table ModifiArme

C L’interaction entre la base de données et le site web :

I La page d'inscription :

Pour les inscriptions des nouveaux utilisateurs on récupère les données saisie dans les formulaire d’inscription , après un traitement des information saisie et vérification de ces dernières notamment une prévention contre les injection sql , on vérifie que aucun utilisateur est déjà inscrit avec les donné saisie , puis on crée un nouveaux compte .

```
= "SELECT * FROM compte;";
```

Figure 39: Requête sql permettant l’extraction des données de la base de donné

```
"INSERT INTO compte(idcompte ,nomc,datecreation,serveur ,email,mdp,solde )
        VALUES('$idcompte', '$nomc','$datecreation',  '$serveur', '$email', '$mdp', '$solde') ;";
```

Figure 40: Requête sql permettant la création d'un nouveaux comptes

	idcompt [PK] ch	nomc character varyin	datecrea date	serveur characte	recherche xml	email character varying (50)	mdp character (128)	solde intege	nomguilde character varying (30)
1	1	EtienneBlanc...	2020-...	EUR	NULL	EtienneBlanchard1@...	158205468896bc3b78273f9...	100	UWMOIKAKFRORSOUZZ
2	10 ...	LeroyHuet	2020-...	EUR	NULL	LeroyHuet10@email....	d3f3619fa657173e0311728...	100	FQRQNVIBRAOSQQ

Figure 41: illustration des données de la table compte

II La page de connexion :

Après vérification des données saisie par l'utilisateur sur le formulaire, hachage du mot de passe saisie et suppression des caractère html pour prémunir contre les injection sql . A l'aide d'une requête sql on vérifie si il existe dans notre base de donné un utilisateur correspondant a l'adresse mail saisie ainsi que le mot de passe correspondant . Si ces deux dernière condition sont vérifie alors on redirige l'utilisateur vers la page d'accueil (espace personnelle).

```
"SELECT idcompte  FROM compte WHERE email = '$email' AND mdp = '$mdp' ; "
```

Figure 42: requête sql permettant la récupération idcompte utilisateur

	idcompte [PK] char	nomc character varying	datecreation date	serveur character vary	recherchep xml	email character varyin	mdp character (128)	solde integer	nomguilde character varying (30)
1	1	EtienneBlancha...	2020-11-19	EUR	NULL	EtienneBlanc...	158205468896bc3b78273f9e96b5139857e00...	100	UWMOIKAKFRORSOUZZ
2	10	LeroyHuet	2020-06-09	EUR	NULL	LeroyHuet10...	d3f3619fa657173e031172886343c202c5adc6...	100	FQRQNVIBRAOSQQ
3	11	PerrinParis	2019-02-23	EUR	NULL	PerrinParis11...	ba4e6b6ddac3a8d3bd77dc0b560e96e0a46b4...	100	[null]
4	12	MassonChevall...	2019-04-12	NA	NULL	MassonChev...	13f65b1d12c31a8a92dee2a76664d41bb8676...	100	[null]

Figure 43: illustration de la table compte avec ces différents attributs

III La page d'accueil :

Sur la page d'accueil de notre site web on trouve des information relative au compte d'utilisateur (titulaire du compte , numéro de compte , solde de compte) ces informations sont extraite de notre base de donné a l'aide d'une requête sql prenant comme attributs l'id de l'utilisateur courant .elle sont extraite de la table compte (Figure 41).

```
"SELECT * FROM  compte WHERE idcompte= '$id';";
```

Figure 44: requête sql permettant l'extraction des données de utilisateur a l'aide de l'id

IV La page de classement :

A l'aide d'une requet sql on récupère le nom du personnage (nomper) sa position dans le classement (position) de la saison choisie par l'utilisateur .cette requête contient une joiture entre 3 tables :

- la table personnage .
- la table Estclasse.
- la table classement.

```
"select nomper, position , saison from
(SELECT nomper, idclassement  FROM personnage p, estclasse E WHERE p.idPer = E.idper) as tab,
classement as c Where tab.idclassement = c.idclassement   and saison = '$saison';";
```

Figure 45: Requête sql permettant l'extraction nom ,position,saison de utilisateur

V La page de Compte :

Dans une première partie de cette page nous avons réussi a afficher

la date de création du compte , le serveur , l'e-mail de l'utilisateur , le solde et le nom de Guilde .

Grace a la requête sql ci dessous, on se servant de l'id de l'utilisateur .

```
$requete = "SELECT nomC,dateCreation,serveur, eMail, solde, nomGuilde  FROM compte WHERE idcompte= '$id';";
```

Figure 46: Requête sql permettant l'extraction nom, date création ,email,solde et nomGuilde

Puis dans une seconde partie nous avons générée le nom du personnage , cela grâce a une requête sql avec une jointure entre la table personnage et la table compte .

```
$requete1 = "SELECT  nomper FROM personnage as pr, compte as c WHERE c.idcompte = pr.idcompte and c.idcompte= '$id';";
```

Figure 47: Requête sql permettant l'extraction nom personnage

VI La page de Équipement :

Dans une première partie , sur la page équipement nous avons réussi a extraire l'ensemble des arme détenu par chaque jouer a l'aide de la requête a 2 jointure entre la table Équipement et la table arme .

```
"select nomC, niveau, attaque, critChance, degatCrit, vitesseAttaque  from
Equipement AS Eq ,  Arme AS Ar WHERE  Eq.idEquipement = Ar.idArme  and idcompte = '$id';";
```

Figure 48: Requête sql permettant l'extraction nom arme,niveau,attaque,crichance,degatCrit,vitesseAttaque

Nous avons également réussi a extraire l'ensemble des armure détenu par chaque jouer a l'aide de la requête a 2 jointure entre la table Équipement et la table armure .

```
$requete = "select nomC, niveau,  defence from Equipement AS Eq ,
            Armure AS Ar WHERE  Eq.idEquipement = Ar.idArmure and idcompte = '$id';";
```

Figure 49: Requête sql permettant l'extraction nom armure,niveau,defence

VII La page acheter des objets :

Sur la page AcheterObjet.php nous avons la possibilité d'acheter des armes ou bien des armures nous avons aussi la possibilité de choisir le nombre de modificateurs, le prix maximum de l'objet que nous voulons acheter .

cette requête est une jointure entre deux tables, Équipement et arme pour ce qui concerne la table s est une représentation de deux attributs id arme et nombre de modificateur cette table est obtenue par une requête imbriquée dans cette dernière nous avons essayé de récupérer tous les identifiants de table arme qui respectent les conditions citées dans la clause WHERE après pour compter les nombres de modificateurs nous avons utilisé une fonction d'agrégation "count" ensuite pour les variables $prix et $modificateur ce sont deux variables qui présentent le prix maximum et le nombre de modificateurs choisi par l'utilisateur

```
"SELECT nomc, niveau, attaque,critchance, degatcrit,vitesseattaque,prix,a.idArme,count
 FROM Equipement e, Arme a ,
 (SELECT s.idArme,count FROM (SELECT m.idArme, COUNT (*) as count
 FROM  modifiarme m GROUP BY m.idArme) as s WHERE s.count>='$modificateur') as b
 WHERE e.idEquipement= a.idArme AND a.idArme=b.idArme AND e.prix>0 AND e.prix<'$prix' and  e.idcompte != '$id' ;";
```

Figure 50: requête sql permettant l'extraction du nomc , niveau ,attaque...

2 La partie réseau :

A Fonctionnement du serveur réseau:

Le serveur réseaux est réalisé en java. Son but est de d'offrir un moyen simple d'effacer des comptes de la base de données. Dans notre contexte, il s'agit d'un outil pour bannir définitivement un compte d'un joueur. Le serveur fonctionne de manière similaire à un automate:
Il est composé de plusieurs états (état initial, sélection de l'id d'un compte...). Dans une utilisation normale et sans problèmes, le serveur s'utilise de la manière suivante:

1. attente de notification de sélection d'un compte par le client(état 1)
2. attente de sélection de l'id d'un compte par le client, puis affichage du compte sélectionné (état 2)
3. attente de demande de l'opération de suppression par client, suivi d'un affichage de toutes les tables qui seront supprimées (état 3)
4. attente de la confirmation de la suppression d'un compte, puis suppression du compte et de tout ce qu'il possède (personnages, équipement, etc) (état 4)

De plus, le serveur peut envoyer des informations potentiellement utiles pour déboguer, tel que l'état du serveur réseaux , la connexion serveur base de données... A tout moment il est possible de revenir à l'état de base (attente de notification de sélection d'un compte) via la commande cancel. Enfin, si l'opération que doit faire le serveur ou la commande envoyée par le client est incorrecte, le serveur ne change pas d'état.

B Fonctionnement du client réseau :

Le client réseau est très léger et on pourrait se limiter à utiliser netcat : Le client réseaux ne fait "que" lire les données tapées sur pas la console par l'utilisateur et les retransmet au serveur.

C Protocole Applicatif :

Dans ce qui suit, les commandes écrites en noir sont des commandes ou résultats exacts, alors que les commandes ou résultat en rouge indique ce qu'il fait écrire ou ce qu'on l'on reçoit (exemple : si il y a écrit "entier positif" il faut donner 1 ou 2 ou 3... De plus, on omettra dans la plupart des cas les messages d'acquittement pour ne pas charger inutilement le diagramme. Dans un premier temps, voici le dialogue de connexion client-serveur. Il s'agit d'une connexion TCP classique, où l'on rajoute en plus un envoi du client au serveur son nom.

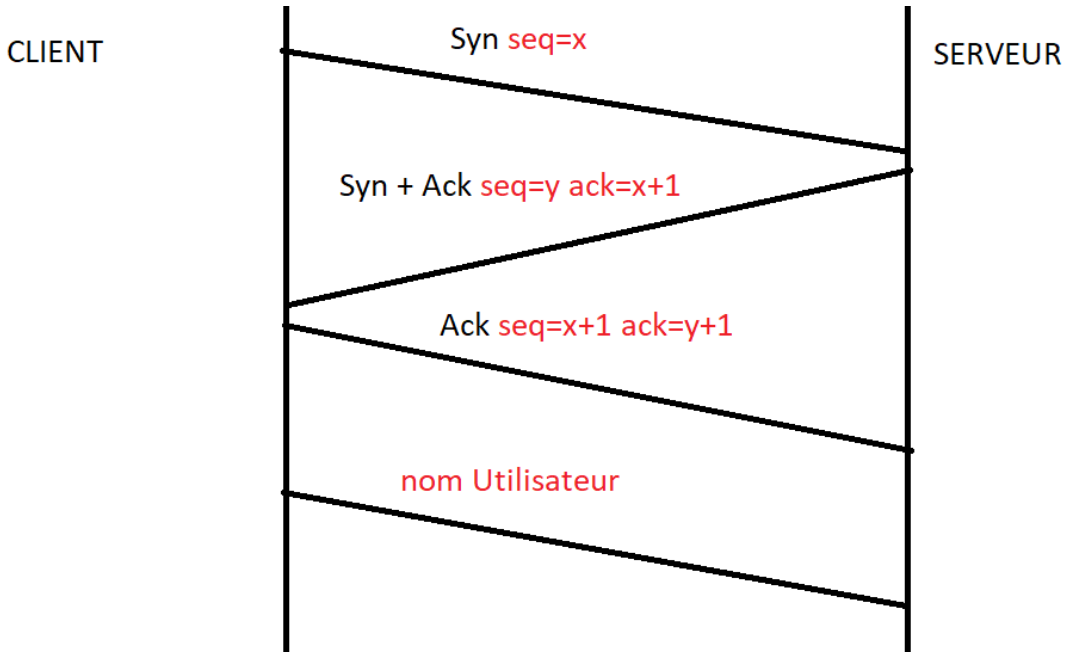

Figure 51: début de connexion

De même pour la déconnexion, qui ne peut être lancée qu'au niveau du client, lorsque l'utilisateur entre "end" sur la console. Il s'agit des quatre phases d'une fermeture de connexion TCP classique.

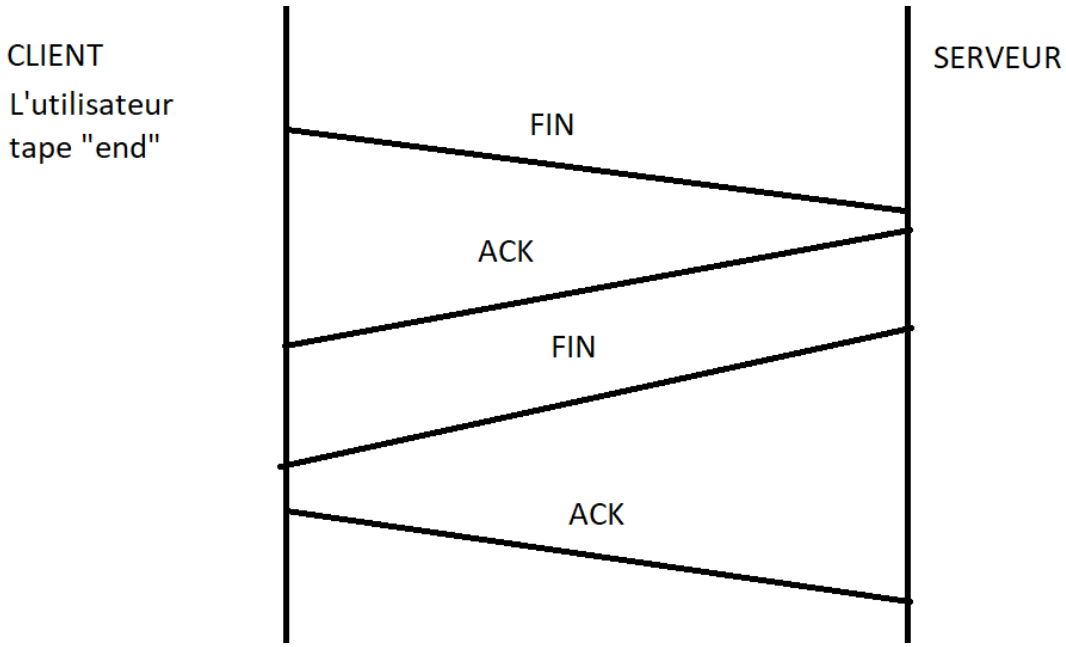

Figure 52: Fermeture de connexion

Voici un exemple de dialogue ou tout se passe bien. On a omis l'étape de connexion et de déconnexion.

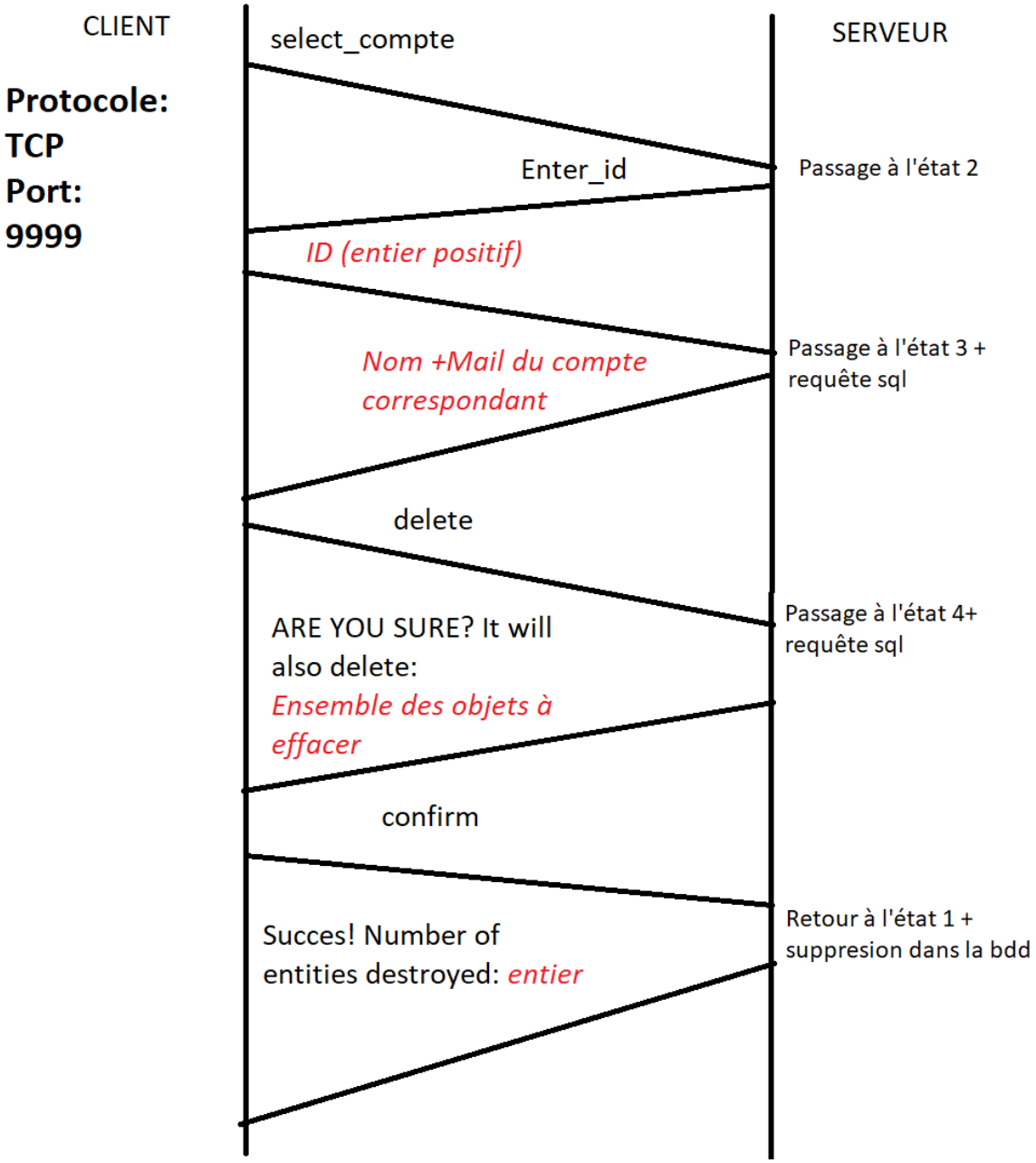

Figure 53: Exemple de dialogue classique

Ici, on montre l'usage de deux commandes (debug et cancel)pouvant être utiliser à n'importe quel moment.

Le client réseau est très léger et on pourrait se limiter à utiliser netcat: Le client réseaux ne fait la plupart du temps "que" lire les données tapées sur pas la console par l'utilisateur et les retransmet au serveur. Cependant, lorsque que l'utilisateur est sensé envoyer un entier, il vérifie quand même que ce qui est saisi est un nombre.

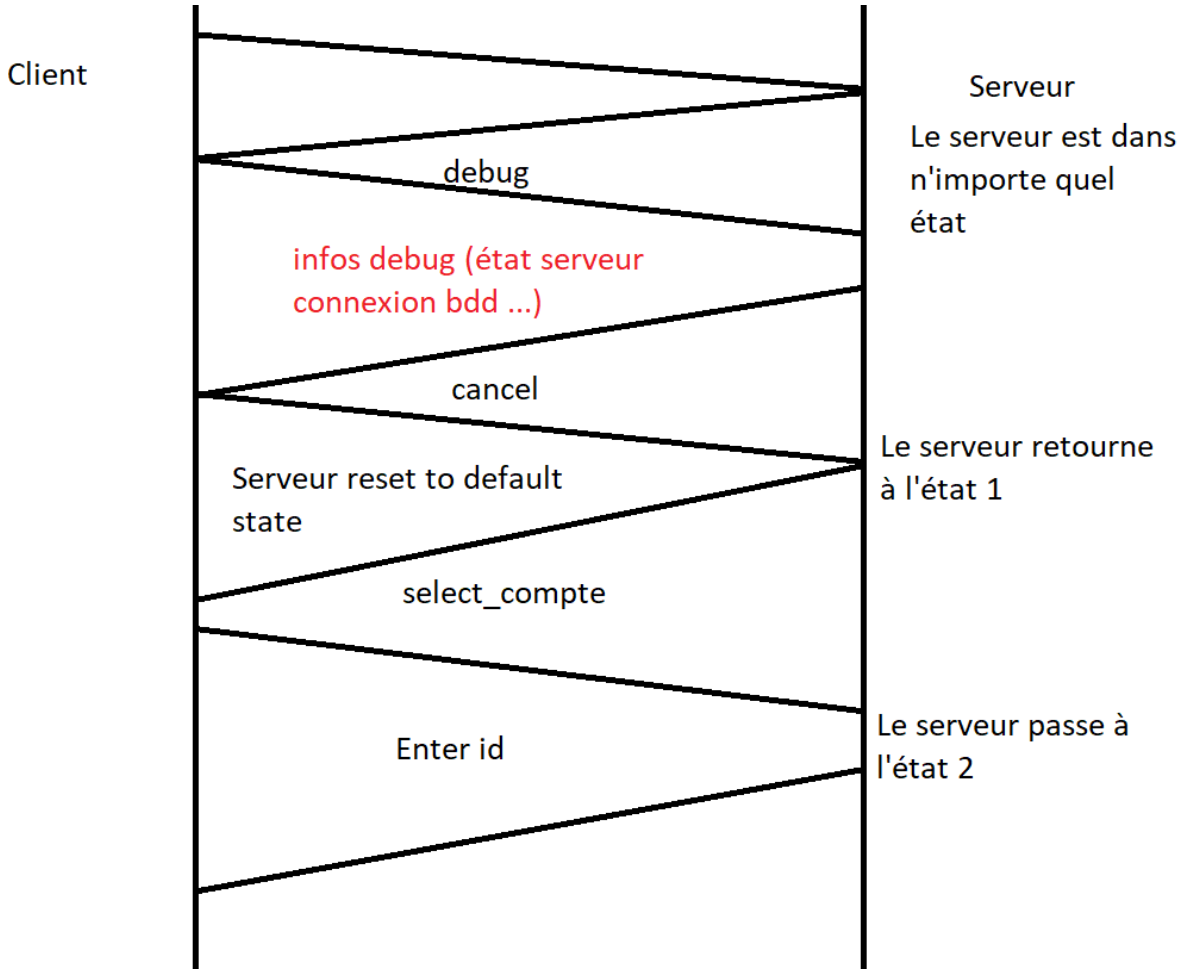

Figure 54: Autres commandes disponibles

Dans cette partie on montre l'ensemble des résultats possibles pour la réponse à la question Enter_id du serveur. Ici l'on suppose que l'utilisateur commence par envoyer quelque chose qui n'est pas un nombre, puis un id qui n'est pas présent dans la base de données. Finalement l'utilisateur entre un id valide.

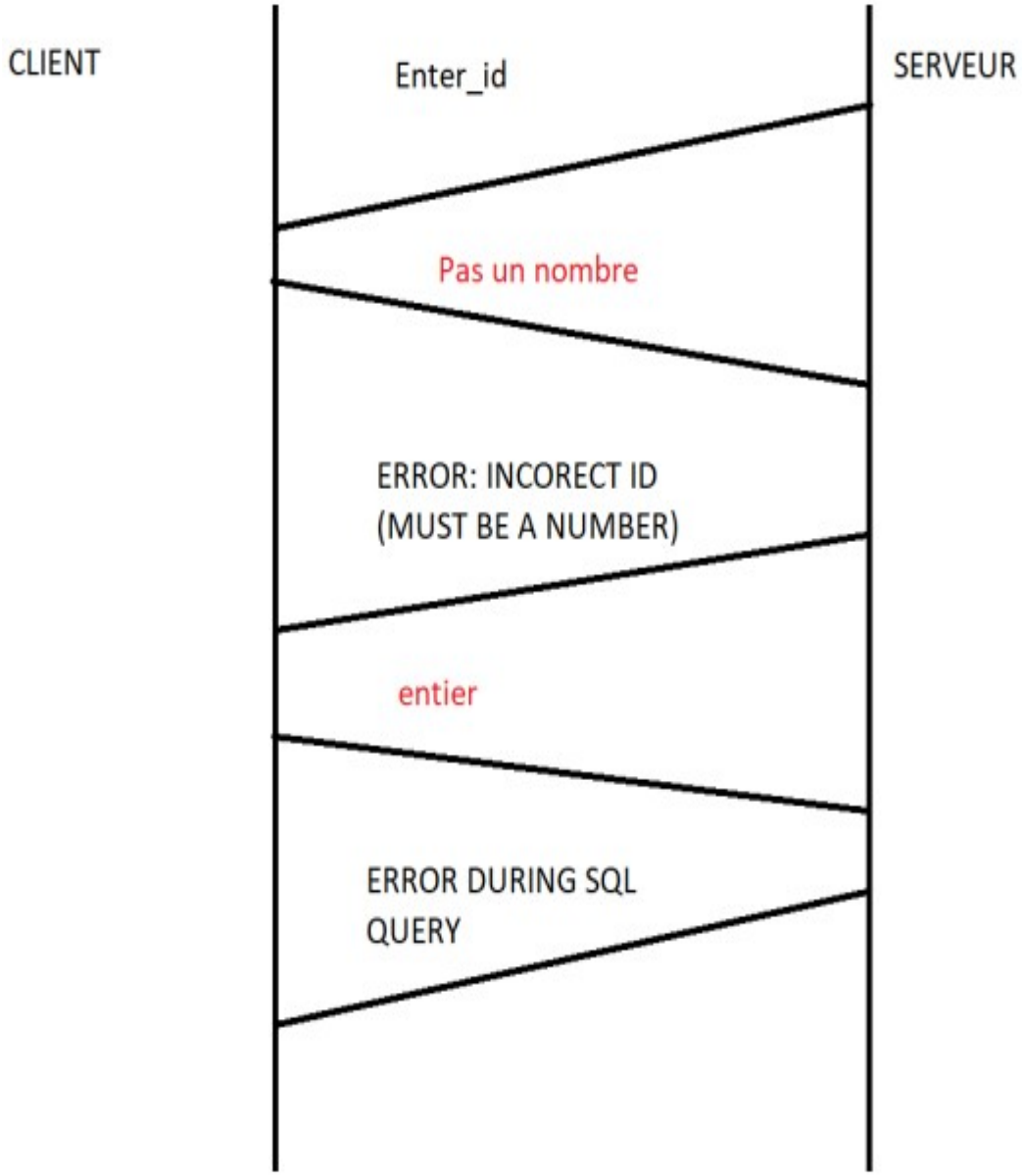

Figure 55: Message d’erreur suivant la commande "Enter_id"

Maintenant on s’intéresse aux erreurs des commandes "delete" et "confirm" lorsqu'elles sont utilisées au moment où il le faut. Elles ne peuvent renvoyer qu’une seule erreur indiquée ci dessous :

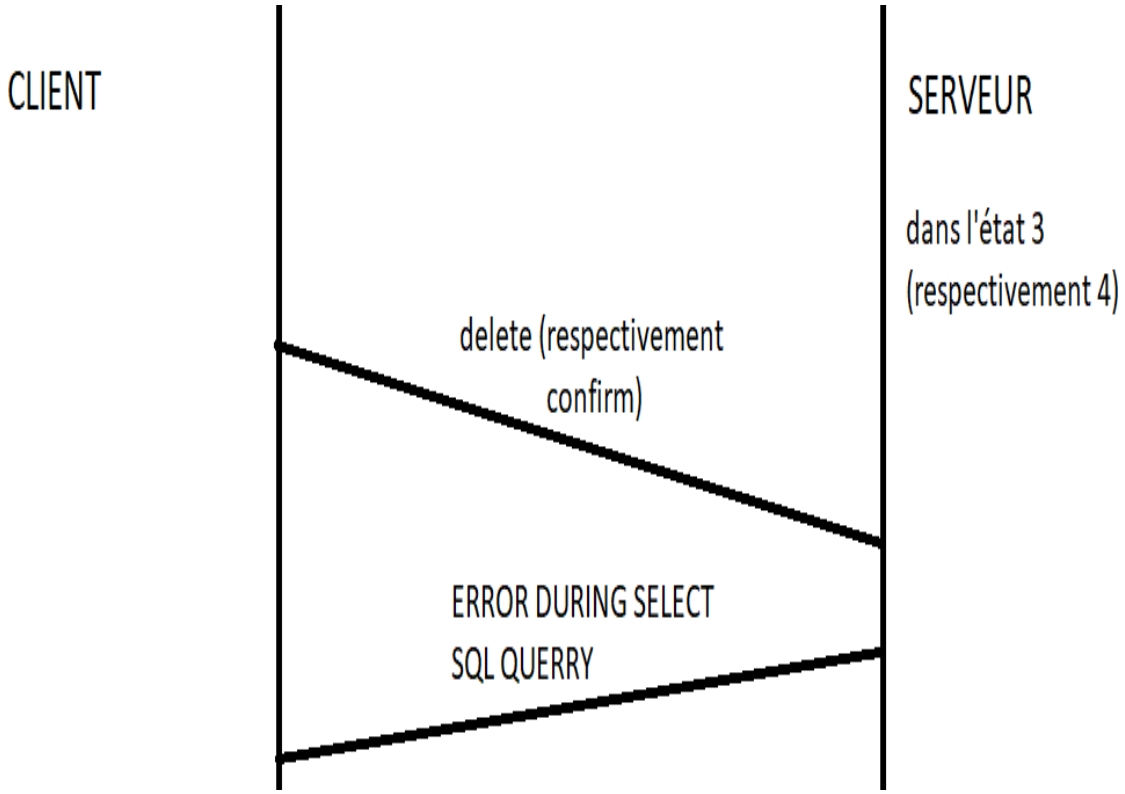

Figure 56: Message d’erreur suivant la commande "delete" ou "confirm"