

Table des matières

Exercice 1 : Chargement du fichier dans une table unique	2
1-création d'une base de données :	3
2-Se connecter à la base de données :	3
3-Création du rôle avec mdp :	3
4- Accordez tous les droits nécessaires pour rendre le schéma <i>etudiants</i> opérationnel dans la base pise	3
Creation des tables :	4
Import des fichiers csv :	7
Exercice 2 : nettoyage des colonnes	9
1. Après avoir exécuté les requêtes ci-dessous, dites quelles actions ont-elles sur les données et la structure de la table ECRITURE :	9
2. Proposez une ou plusieurs requêtes qui changent le type de données des colonnes HT, TVA, TTC de TEXT à DECIMAL	9
3. Proposez une ou plusieurs requêtes qui change le type de données des colonnes IDT_MODE_REG, IDT_STATUT et IDT_SS_CATEGORIE de TEXT à INTEGER . Que constatez-vous sur le cas de la colonne IDT_SS_CATEGORIE ? Comment y remédier ?	10
4. Supprimez toutes les écritures qui n'ont pas de DATE puis renommez la colonne DATE en DT_ECRITURE . Enfin proposez une requête qui change le type de DT_ECRITURE de TEXT à DATE	11
5. Supprimez toutes les écritures qui n'ont pas de NUMERO	12
6. Exécutez et commentez la requête suivante :	12
7. Changez le type de données de la colonne EXERCICE_COMPTABLE de TEXT à INTEGER	14
Exercice 3 : création des contraintes sur les colonnes de tables	15
Exercice 4 : vers l'établissement du bilan comptable	19
1. Proposer une requête SQL qui liste le nombre d'écritures ainsi que la somme des montants HT, TVA, TTC des écritures par année comptable. Cette liste doit être triée sur l'année de l'exercice comptable, de la plus récente à la plus ancienne.	19
2. Proposer une requête SQL qui affiche le montant moyen TTC des écritures de catégorie « Fonctionnement Entreprise »	19
3. Proposer une requête SQL qui affiche le nombre, la somme HT, la somme TTC par type d'opération	21
4. Proposez une requête SQL qui permet d'afficher le bilan comptable de l'année 2022 sous la forme (ne pas tenir compte des montants affichés, seule la forme compte) :	22
5. Commenter la requête ci-dessous. Que vous inspire le résultat obtenu :	24
Exercice 5 : Export des données	26

Exercice 1 : Chargement du fichier dans une table unique

- Générer un fichier CSV à partir des données de l'onglet ECRITURE du fichier Excel fourni
- Créer la table centrale **ECRITURE** avec toutes les colonnes de type **TEXT**
- Importer dans la table centrale créée l'intégralité du fichier CSV généré
- Créer et alimenter les autres tables

Nettoyage du fichier Excel :

Afin de débiter ce projet, il est nécessaire de mettre en ordre le fichier Excel et donc de le nettoyer correctement.

D'après notre réflexion, pour le nettoyage, plusieurs étapes se sont imposées.

La première étape est de fusionner les onglets en un seul et unique onglet écriture. Pour ce faire il convient de reprendre les informations des autres onglets en faisant correspondre par rapport à la configuration de l'onglet écriture. Toutes les informations sont à récupérer à l'exception des cellules, lignes ou colonnes comportant des calculs (par exemple des sommes ou totaux).

Il est important de converser la mise en page de l'onglet écriture et donc de faire correspondre les nouvelles informations importées.

Il faut ensuite défusionner les cellules. Cela est dans le but de ne pas retrouver des lignes en parties vides lors de la transformation en csv mais également lors de l'import des données.

Il a donc été nécessaire de rechercher toutes les cellules fusionnées et ce dans toutes les colonnes.

Dans certains cas la défusion de cellule a engendré une nouvelle ligne tandis que dans d'autres cas cela a engendré une nouvelle ligne vide qu'il a donc fallu supprimer.

Une fois tout le fichier Excel nettoyé des cellules fusionnées il convient de traiter les potentiels doublons que comporte la colonne n° de pièce. Cela sera nécessaire pour ne pas avoir deux fois un numéro de facture mais dont les montants et type seront différents.

Ensuite, il est nécessaire de supprimer les lignes comportant des calculs comme par exemple les lignes de calculs de fin d'exercice. Il en est de même avec certaines cellules qui sont en dehors du tableau avec des formules de résultats vis-à-vis de plusieurs cellules.

Enfin, une harmonisation de certains noms a été nécessaire en procédant par tri de catégorie dans chaque colonne. Cela a permis de déceler des erreurs de dates avec notamment des dates impossibles. Il est donc nécessaire d'uniformiser le tout. Il convient de faire de même avec chaque colonne. De nombreux noms ont été modifiés dans la colonne type avec par exemple des différences comme péage et péage autoroute. L'uniformisation de tous les noms qui correspondent au même type a permis de réduire le nombre de type mais aussi d'avoir une certaine concordance pour l'import des données. Cela a été également le cas pour des types identiques mais dont des fautes d'orthographe engendrent des erreurs (dans le sens où deux types différents existent alors).

Cette tentative de nettoyage a été procédée pour chaque colonne, cellule (par exemple cela a permis de déceler un montant qui comportait une faute de frappe avec un ç à la place du 9).

Pour conclure, il est nécessaire d'ajouter des colonnes `idt_mode_reg`, `idt_statut` et `idt_ss_categorie`. Une fois ces colonnes ajoutées et remplies, le fichier est terminé et prêt à être enregistré puis converti au format csv pour importation.

Ensuite nous sommes passés à la création d'une base de données pour entamer le travail sur Postgrés.

1-création d'une base de données :

```
create database pise ;
```

2-Se connecter à la base de données :

Afin de se connecter à la base de données PISE en tant que DBA :

```
psql -h localhost -p 5432 -U postgres -d pise
```

3-Création du rôle avec mdp :

Création du schéma `etudiants` avec le mot de passe `etu_pise_2023`

4- Accordez tous les droits nécessaires pour rendre le schéma `etudiants` opérationnel dans la base `pise`

```
CREATE ROLE étudiants LOGIN PASSWORD 'etu_pise_2023';
```

```
CREATE SCHEMA IF NOT EXISTS étudiants AUTHORIZATION étudiants;
```

```
GRANT CONNECT ON DATABASE PISE TO étudiants;
```

```
GRANT USAGE ON SCHEMA étudiants TO étudiants;
```

```
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA étudiants TO étudiants;
```

On crée le tablespace dans le tablespace data_pise dans le repertoire C:\temp :

```
CREATE TABLESPACE data_pise  
LOCATION  
'C:\temp' ;
```

Attribution comme propriétaire de tablespace le compte etudiants :

```
ALTER TABLESPACE data_pise  
OWNER TO etudiants ;
```

```
pise=# CREATE TABLESPACE data_pise  
pise=# LOCATION  
pise=# 'C:\temp' ;  
CREATE TABLESPACE  
pise=# ALTER TABLESPACE data_pise  
pise=# OWNER TO etudiants ;  
ALTER TABLESPACE  
pise=#
```

Creation des tables :

Se connecter en tant que schema pour créer nos six tables :

```
psql -h localhost -p 5432 -U etudiants -d pise
```

Création de la table ecriture :

```
pise=> CREATE TABLE ECRITURE  
pise-> (  
pise(> DATE TEXT,  
pise(> EXERCICE TEXT,  
pise(> N°PIECE TEXT,  
pise(> NOM_FICHER TEXT,  
pise(> LIBELLE TEXT,  
pise(> TYPE TEXT,  
pise(> TTC TEXT,  
pise(> TVA TEXT,  
pise(> HT TEXT,  
pise(> MODE_RGLT TEXT,  
pise(> ETAT_STATUT TEXT,  
pise(> IDT_MODE_REG TEXT,  
pise(> IDT_STATUT TEXT,  
pise(> IDT_SS_CATEGORIE TEXT  
pise(> ) TABLESPACE data_pise;  
CREATE TABLE
```

```
CREATE TABLE ECRITURE
(
DATE TEXT,
EXERCICE TEXT,
N°PIECE TEXT,
NOM_FICHER TEXT,
LIBELLE TEXT,
TYPE TEXT,
TTC TEXT,
TVA TEXT,
HT TEXT,
MODE_RGLT TEXT,
ETAT_STATUT TEXT,
IDT_MODE_REG TEXT,
IDT_STATUT TEXT,
IDT_SS_CATEGORIE TEXT
) TABLESPACE data pise;
```

Création de la table catégorie :

```
pise=> CREATE TABLE CATEGORIE
pise-> (
pise(> IDT_CATEGORIE INTEGER,
pise(> LIBELLE_CATEGORIE VARCHAR(50),
pise(> IDT_TYPE_OPE INTEGER
pise(>
pise(> ) TABLESPACE data_pise;
CREATE TABLE
pise-> CREATE TABLE TYPE OPERATION
```

```
CREATE TABLE CATEGORIE
(
  IDT_CATEGORIE INTEGER,
  LIBELLE_CATEGORIE VARCHAR(50),
  IDT_TYPE_OPE INTEGER
) TABLESPACE data pise;
```

Création de la table type_operation :

```
pise=> CREATE TABLE TYPE_OPERATION
pise-> (
pise(> IDT_TYPE_OPE INTEGER,
pise(> LIBELLE VARCHAR(20)
pise(>
pise(> ) TABLESPACE data_pise;
CREATE TABLE
```

```
CREATE TABLE TYPE_OPERATION
(
IDT_TYPE_OPE INTEGER,
LIBELLE VARCHAR(20)

) TABLESPACE data_pise;
```

Création de la table statut :

```
pise=> CREATE TABLE STATUT
pise-> (
pise(> IDT_STATUT INTEGER,
pise(> LIBELLE_STATUT VARCHAR(20)
pise(>
pise(> ) TABLESPACE data_pise;
CREATE TABLE
```

```
CREATE TABLE STATUT
(
IDT_STATUT INTEGER,
LIBELLE_STATUT VARCHAR(20)

) TABLESPACE data_pise;
```

Création de la table sous_categorie :

```
pise=> CREATE TABLE SOUS_CATEGORIE
pise-> (
pise(> IDT_SS_CATEGORIE INTEGER,
pise(> LIBELLE_SS_CATEGORIE TEXT,
pise(> IDT_CATEGORIE INTEGER
pise(>
pise(> ) TABLESPACE data_pise;
CREATE TABLE
```

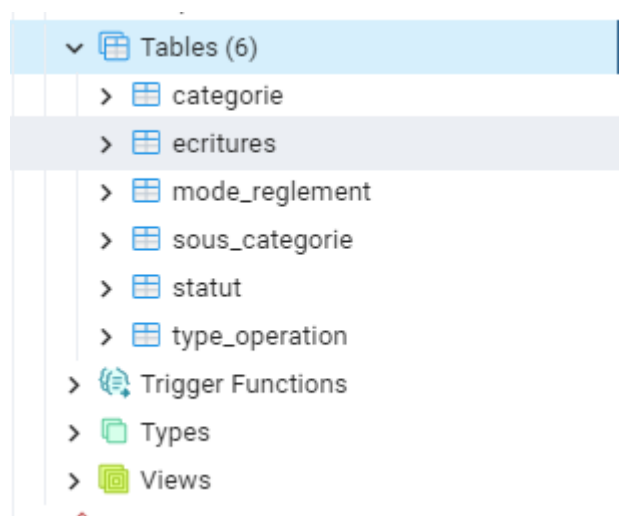
```
CREATE TABLE SOUS_CATEGORIE  
(  
  IDT_SS_CATEGORIE INTEGER,  
  LIBELLE_SS_CATEGORIE TEXT,  
  IDT_CATEGORIE INTEGER  
  
  ) TABLESPACE data_pise;
```

Création de la table mode_reglement :

```
pise=> CREATE TABLE MODE_REGLEMENT  
pise-> (  
pise(> IDT_MODE_REG INTEGER,  
pise(> LIBELLE_MODE_REG VARCHAR(20)  
pise(>  
pise(>  
pise(> ) TABLESPACE data_pise;  
CREATE TABLE
```

```
CREATE TABLE MODE_REGLEMENT  
(  
  IDT_MODE_REG INTEGER,  
  LIBELLE_MODE_REG VARCHAR(20)  
  
  ) TABLESPACE data_pise;
```

Résultat :



Import des fichiers csv :

Se connecter en tant que dba pour importer les fichiers csv :

```
psql -h localhost -p 5432 -U postgres -d Pise
```

Table ecriture :

```
COPY ETUDIANTS.ECRITURE (DATE, EXERCICE, N°PIECE, NOM_FICHIER,  
LIBELLE, TYPE, TTC,TVA,HT,MODE_RGLT, ETAT_STATUT, IDT_MODE_REG,  
IDT_STATUT, IDT_SS_CATEGORIE)  
FROM 'c:/temp/ECRITURE.CSV'  
DELIMITER ';' csv header ;
```

Table CATEGORIE :

```
COPY ETUDIANTS.CATEGORIE (IDT_CATEGORIE, LIBELLE_CATEGORIE,  
IDT_TYPE_OPE)  
FROM 'c:/temp/CATEGORIE.CSV'  
DELIMITER ';' csv header ;
```

Table TYPE_OPERATION :

```
COPY ETUDIANTS.TYPE_OPERATION (IDT_TYPE_OPE, LIBELLE)  
FROM 'c:/temp/TYPE_OPERATION.CSV'  
DELIMITER ';' csv header ;
```

Table STATUT :

```
COPY ETUDIANTS.STATUT (IDT_STATUT, LIBELLE_STATUT)  
FROM 'c:/temp/STATUT.CSV'  
DELIMITER ';' csv header ;
```

Table SOUS_CATEGORIE :

```
COPY ETUDIANTS.SOUS_CATEGORIE (IDT_SS_CATEGORIE,  
LIBELLE_SS_CATEGORIE, IDT_CATEGORIE)  
FROM 'c:/temp/SOUS_CATEGORIE.CSV'  
DELIMITER ';' csv header ;
```

Table MODE_REGLEMENT:

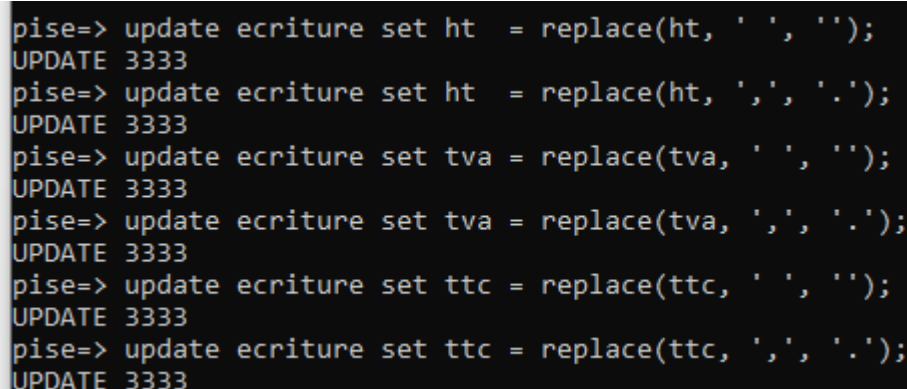
```
COPY ETUDIANTS.MODE_REGLEMENT (IDT_MODE_REG, LIBELLE_MODE_REG)  
FROM 'c:/temp/MODE_REGLEMENT.CSV'  
DELIMITER ';' csv header ;
```

Après cette étape, les données seront en base et le reste des exercices se déroulent dans la table de données à coup de requêtes SQL.

Exercice 2 : nettoyage des colonnes

1. Après avoir exécuté les requêtes ci-dessous, dites quelles actions ont-elles sur les données et la structure de la table ECRITURE :

```
update ecriture set ht = replace(ht, ' ', '');  
update ecriture set ht = replace(ht, ',', '.');  
update ecriture set tva = replace(tva, ' ', '');  
update ecriture set tva = replace(tva, ',', '.');  
update ecriture set ttc = replace(ttc, ' ', '');  
update ecriture set ttc = replace(ttc, ',', '.');
```



```
pise=> update ecriture set ht = replace(ht, ' ', '');  
UPDATE 3333  
pise=> update ecriture set ht = replace(ht, ',', '.');  
UPDATE 3333  
pise=> update ecriture set tva = replace(tva, ' ', '');  
UPDATE 3333  
pise=> update ecriture set tva = replace(tva, ',', '.');  
UPDATE 3333  
pise=> update ecriture set ttc = replace(ttc, ' ', '');  
UPDATE 3333  
pise=> update ecriture set ttc = replace(ttc, ',', '.');  
UPDATE 3333
```

Ces requêtes ont deux effets sur les données et la structure de la table. Dans un premier temps, cela supprime les espaces entre les chiffres et ensuite cela modifie les virgules en points.

2. Proposez une ou plusieurs requêtes qui changent le type de données des colonnes HT, TVA, TTC de TEXT à DECIMAL.

Changement du type de données de la colonne HT en DECIMAL :

```
ALTER TABLE etudiants.ecriture ALTER COLUMN HT TYPE decimal  
USING ht::decimal;
```

Changement du type de données de la colonne TVA en DECIMAL :

```
ALTER TABLE etudiants.ecriture ALTER COLUMN TVA TYPE  
decimal USING TVA::decimal;
```

Changement du type de données de la colonne TTC en DECIMAL :

```
ALTER TABLE etudiants.ecriture ALTER COLUMN ttc TYPE  
decimal USING ttc::decimal;
```

```

pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN HT TYPE decimal USING ht::decimal;
ALTER TABLE
pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN HT TYPE decimal USING ht::decimal;
ALTER TABLE
pise=>
pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN TVA TYPE decimal USING TVA::decimal;
ALTER TABLE
pise=>
pise=>
pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN ttc TYPE decimal USING ttc::decimal;
ALTER TABLE
pise=> commit ;

```

ttc numeric	tva numeric	ht numeric
12.54	0.18	12.36
12.54	0.18	12.36
55.96	0.18	55.78
12.54	0.18	12.36
14.36	0.18	14.18
78.16	0.18	77.97
20.13	0.19	19.94
20.13	0.19	19.94
101.07	0.10	101.78

3. Proposez une ou plusieurs requêtes qui change le type de données des colonnes **IDT_MODE_REG**, **IDT_STATUT** et **IDT_SS_CATEGORIE** de **TEXT** à **INTEGER**. Que constatez-vous sur le cas de la colonne **IDT_SS_CATEGORIE** ? Comment y remédier ?

```
ALTER TABLE etudiants.ecriture ALTER COLUMN IDT_MODE_REG
TYPE INTEGER USING IDT_MODE_REG:: INTEGER;
```

```
ALTER TABLE etudiants.ecriture ALTER COLUMN IDT_STATUT TYPE
INTEGER USING IDT_STATUT:: INTEGER;
```

```
ALTER TABLE etudiants.ecriture ALTER COLUMN
IDT_SS_CATEGORIE TYPE INTEGER USING IDT_SS_CATEGORIE::
INTEGER;
```

```

pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN IDT_MODE_REG TYPE INTEGER USING IDT_MODE_REG:: INTEGER;
ALTER TABLE
pise=>
pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN IDT_STATUT TYPE INTEGER USING IDT_STATUT:: INTEGER;
ALTER TABLE
pise=>
pise=> ALTER TABLE etudiants.ecriture ALTER COLUMN IDT_SS_CATEGORIE TYPE INTEGER USING IDT_SS_CATEGORIE:: INTEGER;
ALTER TABLE
pise=> commit;

```

Nous ne constatons rien sur le cas de la colonne IDT_SS_CATEGORIE et nous ne pouvons donc pas y remédier.

idt_mode_reg integer	idt_statut integer	idt_ss_categorie integer
5	[null]	[null]
5	[null]	[null]
5	[null]	[null]
5	[null]	[null]
5	[null]	[null]
5	[null]	[null]
5	[null]	[null]

- Supprimez toutes les écritures qui n'ont pas de DATE puis renommez la colonne **DATE** en **DT_ECRITURE**. Enfin proposez une requête qui change le type de **DT_ECRITURE** de **TEXT** à **DATE**.

Afin de de supprimer les écritures qui n'ont pas de DATE il convient d'exécuter la requête suivant :

```
delete from etudiants.ecriture where DATE is null ;
```

Afin de de changer le type de DT_ECRITURE de TEXT à DATE il convient d'exécuter la requête suivant :

```
ALTER TABLE ecriture ALTER COLUMN DT_ECRITURE TYPE DATE
USING DT_ECRITURE::DATE ;
```

```

pise=> ALTER TABLE ecriture ALTER COLUMN DT_ECRITURE TYPE DATE USING DT_ECRITURE::DATE ;
ALTER TABLE
pise=> commit;

```

	dt_ecriture text	exercice text
1	22/03/2018	2018
2	11/12/2018	2019
3	25/02/2019	2019
4	janv-18	2018



	dt_ecriture date	exercice integer	n°pièce text
1	2012-10-29	2013	Facture N° 1058251
2	2012-11-05	2013	Facture N° 12156205
3	2012-11-02	2013	Facture N° 120006
4	2012-11-02	2013	Facture N° 12307016009512
5	2012-11-14	2013	Facture N° 12319015015168
6	2012-11-15	2013	Facture N° 3452
7	2012-11-28	2013	Facture N° AD0624860
8	2012-12-05	2013	Facture N° 031201212010167
9	2012-12-06	2013	Facture N° 031201212010228

5. Supprimez toutes les écritures qui n'ont pas de NUMERO

Requête permettant la suppression des écritures qui ne possèdent pas de NUMERO :

```
delete from etudiants.ecriture where N°PIECE IS NULL;
```

6. Exécutez et commentez la requête suivante :

```
select distinct exercice from ecriture  
order by 1;
```

```
oise=> select distinct exercice from ecritures
oise-> order by 1;
exercice
-----
2012
2012-2013
2013
2013-2014
2014
2015
2016
2017
2018
2019
2020
2021
2022
(13 lignes)
```

Que constatez-vous ? Corriger les anomalies suivantes :

Nous constatons des anomalies d'années avec des lignes composées de deux années séparées d'un tiret.

2012-2013 en 2012

2013-2014 en 2013

null en 2022

Pour corriger les anomalies il est nécessaire d'exécuter les requêtes suivantes :

```
UPDATE ecriture
SET exercice = '2012'
WHERE exercice = '2012-2013' ;
```

```
UPDATE ecriture
SET exercice = '2013'
WHERE exercice = '2013-2014' ;
```

```
UPDATE ecriture
SET exercice = '2022'
WHERE exercice is NULL ;
```

Après avoir exécuté ces scripts, on a fait un commit.

Exemple :

	dt_ecriture text	exercice text	nøpiece text		dt_ecriture text	exercice text	nøpi text
162	29/11/2013	2013-2014	Node D	→	162	06/12/2013	2014
163	29/11/2013	2013-2014	Node D		163	10/12/2013	2014

```

pise=> UPDATE ecritures
pise-> SET exercice = '2012'
pise-> WHERE exercice = '2012-2013' ;
UPDATE 25
pise=> UPDATE ecritures
pise-> SET exercice = '2013'
pise-> WHERE exercice = '2013-2014' ;
UPDATE 39

```

- Changez le type de données de la colonne **EXERCICE_COMPTABLE** de TEXT à **INTEGER**.

```
ALTER TABLE etudiants.ecriture ALTER COLUMN EXERCICE TYPE
INTEGER USING EXERCICE:: INTEGER;
```

exercice integer	r t
2018	
2019	
2019	
2018	
2018	

Exercice 3 : création des contraintes sur les colonnes de tables

Proposez des requêtes sql pour répondre aux besoins de contraintes suivants :

- Chaque table doit disposer d'une contrainte de clé de primaire conformément au MLD

TABLE ecriture :

```
ALTER TABLE ecriture ADD CONSTRAINT ecriture_PK PRIMARY KEY  
(N°PIECE) USING INDEX TABLESPACE data_pise ;
```

TABLE TYPE_OPERATION :

```
ALTER TABLE TYPE_OPERATION ADD CONSTRAINT TYPE_OPERATION_PK  
PRIMARY KEY (IDT_TYPE_OPE) ;
```

TABLE CATEGORIE ;

```
ALTER TABLE CATEGORIE ADD CONSTRAINT CATEGORIE_PK PRIMARY  
KEY (IDT_CATEGORIE) ;
```

TABLE SOUS_CATEGORIE :

```
ALTER TABLE SOUS_CATEGORIE ADD CONSTRAINT SOUS_CATEGORIE_PK  
PRIMARY KEY (IDT_SS_CATEGORIE) ;
```

TABLE MODE_REGLEMENT :

```
ALTER TABLE MODE_REGLEMENT ADD CONSTRAINT MODE_REGLEMENT_PK  
PRIMARY KEY (IDT_MODE_REG) ;
```

TABLE STATUT:

```
ALTER TABLE STATUT ADD CONSTRAINT STATUT_PK PRIMARY KEY  
(IDT_STATUT) ;
```

- Créer les contraintes de clés étrangères si besoin conformément au MLD :

Table categorie :

```
ALTER TABLE CATEGORIE
ADD CONSTRAINT CATEGORIE_FK
FOREIGN KEY (IDT_TYPE_OPE)
REFERENCES TYPE_OPERATION(IDT_TYPE_OPE) ;
```

```
pise=> ALTER TABLE CATEGORIE
pise-> ADD CONSTRAINT CATEGORIE_FK
pise-> FOREIGN KEY (IDT_TYPE_OPE)
pise-> REFERENCES TYPE_OPERATION(IDT_TYPE_OPE) ;
ALTER TABLE
```

Table sous_categorie :

```
ALTER TABLE SOUS_CATEGORIE
ADD CONSTRAINT SOUS_CATEGORIE_FK
FOREIGN KEY (IDT_CATEGORIE)
REFERENCES CATEGORIE(IDT_CATEGORIE) ;
```

```
pise=> ALTER TABLE SOUS_CATEGORIE
pise-> ADD CONSTRAINT SOUS_CATEGORIE_FK
pise-> FOREIGN KEY (IDT_CATEGORIE)
pise-> REFERENCES CATEGORIE(IDT_CATEGORIE) ;
ALTER TABLE
```

Table ecriture :

```
ALTER TABLE ecriture
ADD CONSTRAINT ecriture_FK_1
FOREIGN KEY (IDT_MODE_REG)
REFERENCES MODE_REGLEMENT(IDT_MODE_REG) ;
```

```
pise-> ADD CONSTRAINT ecriture_FK_1
pise-> FOREIGN KEY (IDT_MODE_REG)
pise-> REFERENCES MODE_REGLEMENT(IDT_MODE_REG) ;
ALTER TABLE
```

```
ALTER TABLE ecriture
ADD CONSTRAINT ecriture_FK_2
FOREIGN KEY (IDT_STATUT)
REFERENCES STATUT(IDT_STATUT) ;
```

```
pise=> ALTER TABLE ecriture
pise-> ADD CONSTRAINT ecriture_FK_2
pise-> FOREIGN KEY (IDT_STATUT)
pise-> REFERENCES STATUT(IDT_STATUT) ;
ALTER TABLE
```

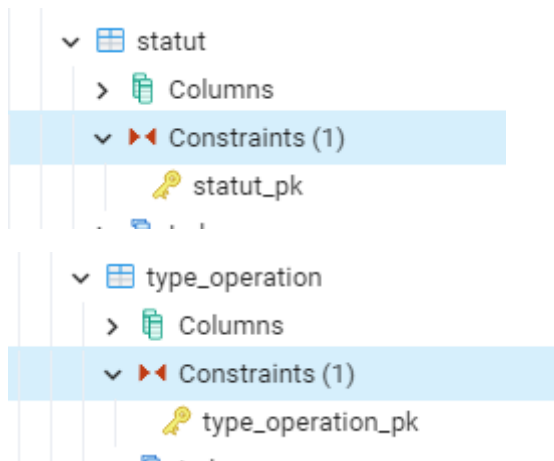


```
ALTER TABLE ecriture  
ADD CONSTRAINT ecriture_FK_3  
FOREIGN KEY (IDT_SS_CATEGORIE)  
REFERENCES SOUS_CATEGORIE (IDT_SS_CATEGORIE);
```

```
pise=> ALTER TABLE ecriture  
pise-> ADD CONSTRAINT ecriture_FK_3  
pise-> FOREIGN KEY (IDT_SS_CATEGORIE)  
pise-> REFERENCES SOUS_CATEGORIE (IDT_SS_CATEGORIE);  
ALTER TABLE
```

Résultats :

▼	📊	categorie
	>	📄 Columns
	▼	🔗 Constraints (2)
		🔑 categorie_fk
		🔑 categorie_pk
		...
▼	📊	ecriture
	>	📄 Columns
	▼	🔗 Constraints (4)
		🔑 ecriture_fk_1
		🔑 ecriture_fk_2
		🔑 ecriture_fk_3
		🔑 ecriture_pk
▼	📊	mode_reglement
	>	📄 Columns
	▼	🔗 Constraints (1)
		🔑 mode_reglement_pk
		...
▼	📊	sous_categorie
	>	📄 Columns
	▼	🔗 Constraints (2)
		🔑 sous_categorie_fk
		🔑 sous_categorie_pk



- Les colonnes suivantes ne doivent pas admettre du null :
 - Au niveau de la table ECRITURE : DATE_ECRITURE, EXERCICE_COMPTABLE, TTC
 - Au niveau de la table MODE_REGLEMENT : LIBELLE_MODE_REG
 - Au niveau de la table STATUT : LIBELLE_STATUT
 - Au niveau de la table CATEGORIE : LIBELLE_CATEGORIE
 - Au niveau de la table SOUS_CATEGORIE : LIBELLE_SS_CATEGORIE

La table ecriture :

```
ALTER TABLE ecriture ALTER COLUMN dt_ecriture SET NOT NULL;  
ALTER TABLE ecriture ALTER COLUMN exercice SET NOT NULL;  
ALTER TABLE ecriture ALTER COLUMN ttc SET NOT NULL;
```

La table Mode_reglement :

```
ALTER TABLE MODE_REGLEMENT ALTER COLUMN libelle_mode_reg  
SET NOT NULL;
```

La table libelle_statut:

```
ALTER TABLE statut ALTER COLUMN libelle_statut SET NOT NULL;
```

La table libelle_categorie:

```
ALTER TABLE categorie ALTER COLUMN libelle_categorie SET  
NOT NULL;
```

La table libelle_ss_categorie:

```
ALTER TABLE sous_categorie ALTER COLUMN  
libelle_ss_categorie SET NOT NULL;
```

A ce stade toutes les tables sont correctes en structure et en données. Les contraintes de clés primaires ainsi que les clés étrangères ont été créées.

Exercice 4 : vers l'établissement du bilan comptable

Il vous est demandé à présent de :

1. Proposer une requête SQL qui liste le nombre d'écritures ainsi que la somme des montants HT, TVA, TTC des écritures par année comptable. Cette liste doit être triée sur l'année de l'exercice comptable, de la plus récente à la plus ancienne.

```
select exercice,
count(*) as "nombre_ecritures",
sum(HT) as "somme_ht",
sum(TVA) as "somme_tva",
sum(TTC) as "somme_ttc"
from ecriture
group by exercice
order by exercice desc;
```

```
pise=> select exercice_comptable,
pise-> count(*) as "nombre_ecritures",
pise-> sum(HT) as "somme_ht",
pise-> sum(TVA) as "somme_tva",
pise-> sum(TTC) as "somme_ttc"
pise-> from ecriture
pise-> group by exercice_comptable
pise-> order by exercice_comptable desc;
```

exercice_comptable	nombre_ecritures	somme_ht	somme_tva	somme_ttc
2022	309	20510.07	2694.98	32242.50
2021	256	19397.85	2885.05	25006.49
2020	212	9892.86	1467.227	30731.88
2019	250	23057.03	4007.66	48655.16
2018	184	11368.71	1762.05	21633.54
2017	335	16474.77	2624.15	69222.29
2016	200	11628.396	2127.934	43080.64
2015	209	14535.15	2743.30	33396.35
2014	138	7990.09	1310.22	26270.12
2013	146	6833.02	1149.16	12421.34
2012	25	194.88	17.84	552.72

2. Proposer une requête SQL qui affiche le montant moyen TTC des écritures de catégorie « Fonctionnement Entreprise »

Pour afficher l'idt_categorie fonctionnement categorie dans la table ecriture :

```
select *
from ecriture ecr
inner join sous_categorie sscat on
(ecr.idt_ss_categorie=
sscat.idt_ss_categorie)
inner join categorie cat on
(sscat.idt_categorie=cat.idt_categorie)
where cat.libelle_categorie='Fonctionnement Entreprise';
```

Dashboard Properties SQL Statistics Dependencies Dependents Processes pise/postgres

pise/etudiants@PostgreSQL 14

Query Query History

```

1 select *
2 from ecriture ecr
3 inner join sous_categorie sscat on
4 (ecr.idt_ss_categorie=
5 sscat.idt_ss_categorie)
6 inner join categorie cat on
7 (sscat.idt_categorie=cat.idt_categorie)
8 where cat.libelle_categorie='Fonctionnement Entreprise'
9

```

Data Output Messages Notifications

	reg	idt_statut integer	idt_ss_categorie integer	idt_ss_categorie integer	libelle_ss_categorie text	idt_categorie integer
1	4	1	25	25	Création entreprise	7
2	4	1	25	25	Création entreprise	7
3	4	1	27	27	Achat KBIS	7
4	1	[null]	29	29	Facture juridique	7
5	3	[null]	27	27	Achat KBIS	7
6	1	[null]	28	28	Dépôt de compte annuels	7
7	5	[null]	26	26	Facture Comptable	7
8	5	[null]	26	26	Facture Comptable	7

Pour afficher le montant moyen TTC des écritures de catégorie « Fonctionnement Entreprise », nous avons utilisé la requête suivante :

```

select
avg (ecr.ttc)
from ecriture ecr
inner join sous_categorie sscat on
(ecr.idt_ss_categorie=
sscat.idt_ss_categorie)
inner join categorie cat on
(sscat.idt_categorie=cat.idt_categorie)
where cat.libelle_categorie='Fonctionnement Entreprise';

```

Query Query History

```

1  select
2  avg (ecr.ttc)
3  from ecriture ecr
4  inner join sous_categorie sscat on
5  (ecr.idt_ss_categorie=
6  sscat.idt_ss_categorie)
7  inner join categorie cat on
8  (sscat.idt_categorie=cat.idt_categorie)
9  where cat.libelle_categorie='Fonctionnement Entreprise'|
10
11

```

Data Output Messages Notifications

	avg numeric	
1	1088.8861904761904762	

3. Proposer une requête SQL qui affiche le nombre, la somme HT, la somme TTC par type d'opération

```

select
top.libelle,
count(ecr.*) as "nombre_ecriture",
sum(ecr.HT) as "somme_ht",
sum(ecr.ttc) as "somme_ttc"
from ecriture ecr
inner join sous_categorie sscat on
(ecr.idt_ss_categorie=
sscat.idt_ss_categorie)
inner join categorie cat
on (sscat.idt_categorie=
cat.idt_categorie)
inner join type_operation top
on (cat.idt_type_ope=
top.idt_type_ope)
group by top.libelle;

```

Query Query History

```

1 select
2 top.libelle,
3 count(ecr.*) as "nombre_ecriture",
4 sum(ecr.HT) as "somme_ht",
5 sum(ecr.ttc) as "somme_ttc"
6 from ecriture ecr
7 inner join sous_categorie sscat on
8 (ecr.idt_ss_categorie=
9 sscat.idt_ss_categorie)
10 inner join categorie cat
11 on (sscat.idt_categorie=
12 cat.idt_categorie)
13 inner join type_operation top
14 on (cat.idt_type_ope=
15 top.idt_type_ope)
16 group by top.libelle;
17

```

Data Output Messages Notifications

	libelle character varying (20)	nombre_ecriture bigint	somme_ht numeric	somme_ttc numeric
1	Décaissement	2139	132440.506	333142.13

4. Proposez une requête SQL qui permet d'afficher le bilan comptable de l'année 2022 sous la forme (ne pas tenir compte des montants affichés, seule la forme compte) :

	LIBELLE_OPE text	somme_ht numeric
1	Décaissement (Crédit)	557066.73
2	Encaissement (Débit)	1189736.14

```
select
top.libelle as "libelle_ope",
sum(ecr.HT) as "somme_ht"
from ecriture ecr
inner join sous_categorie sscat on
(ecr.idt_ss_categorie=
sscat.idt_ss_categorie)
inner join categorie cat
on (sscat.idt_categorie=
cat.idt_categorie)
inner join type_operation top
on(cat.idt_type_ope=
top.idt_type_ope)
where ecr.exercice = '2022'
group by top.libelle;
```

The screenshot shows a SQL query editor interface. The query is as follows:

```
1 select
2 top.libelle as "libelle_ope",
3 sum(ecr.HT) as "somme_ht"
4 from ecriture ecr
5 inner join sous_categorie sscat on
6 (ecr.idt_ss_categorie=
7 sscat.idt_ss_categorie)
8 inner join categorie cat
9 on (sscat.idt_categorie=
10 cat.idt_categorie)
11 inner join type_operation top
12 on(cat.idt_type_ope=
13 top.idt_type_ope)
14 where ecr.exercice_comptable = '2022'
15 group by top.libelle
```

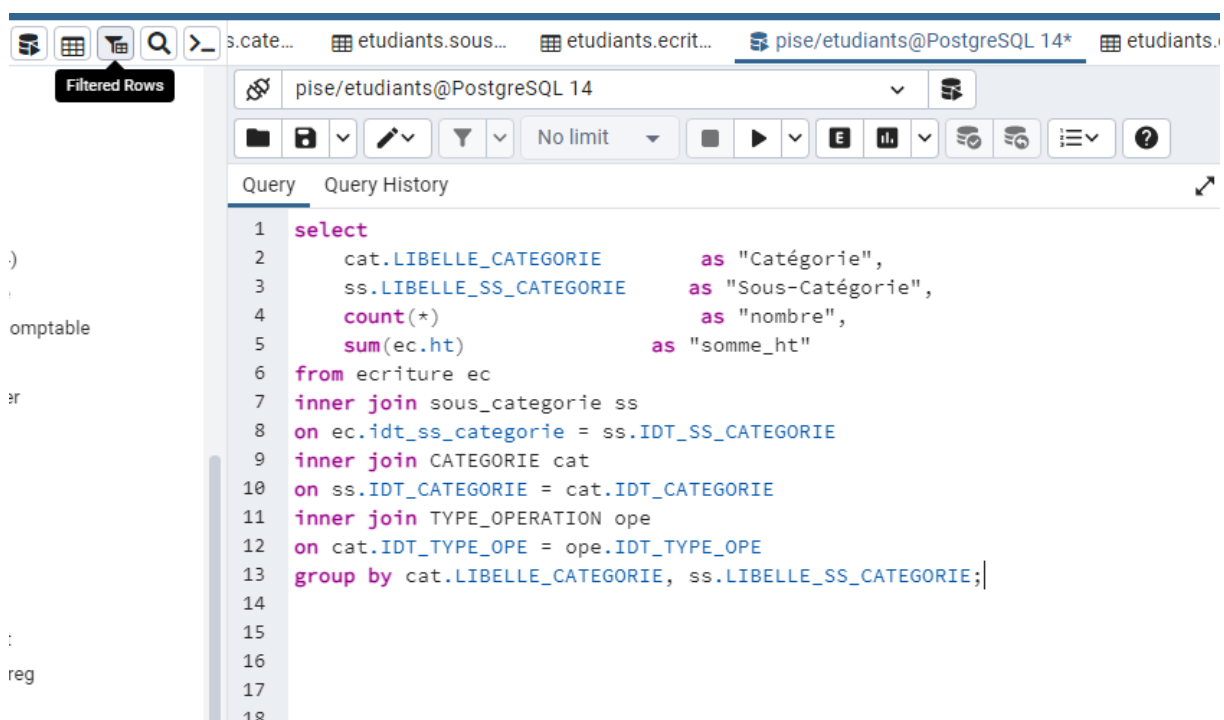
Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table:

	libelle_ope text	somme_ht numeric
1	Décaissement	19803.68

5. Commenter la requête ci-dessous. Que vous inspire le résultat obtenu :

```
select
    cat."LIBELLE_CATEGORIE"           as "Catégorie",
    ss."LIBELLE_SS_CATEGORIE"         as "Sous-Catégorie",
    count(*)                          as "nombre",
    sum(ec.ht)                        as "somme_ht"
from ecriture ec
inner join public."SOUS_CATEGORIE" ss
on ec.idt_ss_categorie = ss."IDT_SS_CATEGORIE"
inner join public."CATEGORIE" cat
on ss."IDT_CATEGORIE" = cat."IDT_CATEGORIE"
inner join public."TYPE_OPERATION" ope
on cat."IDT_TYPE_OPE" = ope."IDT_TYPE_OPE"
group by cat."LIBELLE_CATEGORIE",
ss."LIBELLE_SS_CATEGORIE";
```

Afin de pouvoir exécuter cette requête nous avons du procéder à quelques modifications. Par conséquent, dans notre contexte, il a été nécessaire de supprimer public et les guillemets.



Résultat :

	Catégorie character varying (50)	Sous-Catégorie text	nombre bigint	somme_ht numeric
1	Confort	EDF	77	[null]
2	Trésor Public – Impôts - Taxes	RSI	46	2324.0
3	Assurances	HISCOX	6	[null]
4	Auto	Réparation auto	12	2990.33
5	Consommable Informatique	Abonnement logiciel	30	59.97
6	Fonctionnement Entreprise	Création entreprise	2	179.8
7	Fonctionnement Entreprise	Facture juridique	2	450
8	Auto	Révision auto	8	2469.18
9	Trésor Public – Impôts - Taxes	Amende	1	[null]
10	Trésor Public – Impôts - Taxes	CFE	1	[null]
11	Formation	Achat Livres	7	571.60
12	Fonctionnement Entreprise	Dépôt de compte annuels	3	103.99

On remarque que cette requête affiche les nombres et la somme_ht de tous les sous_catégories.

Exercice 5 : Export des données

Afin qu'une autre équipe soit capable de tester votre travail, il vous est demandé enfin d'exporter dans des fichiers CSV chacune des tables de votre base de données. Il y a en tout 6 tables et l'équipe de testeurs attend de votre part 6 fichiers csv nommés de la manière suivante :

- FIC_TYPE_OPERATION.csv

The screenshot shows the 'Import/Export data - table 'type_operation'' dialog box with the 'General' tab selected. The 'Import/Export' section has 'Import' and 'Export' buttons, with 'Export' being active. The 'Filename' field is set to 'C:\Users\HP\Desktop\sgbd\mini projet\FIC_TYPE_OPERATION.csv'. The 'Format' is set to 'csv' and the 'Encoding' is set to 'WIN1252'. At the bottom, there are buttons for 'Close', 'Reset', and 'OK'.

The screenshot shows the 'Import/Export data - table 'type_operation'' dialog box with the 'Options' tab selected. The 'OID' toggle is off. The 'Header' toggle is on. The 'Delimiter' is set to ';'. The 'Quote' is set to '"'. The 'Escape' is set to '\'. Each option has a descriptive text below it.

• FIC_STATUT.csv

Import/Export data - table 'statut'

General Options Columns

Import/Export

Filename

Format

Encoding

Import/Export data - table 'statut'

General Options Columns

OID ☐

Header ☒

Delimiter
Specifies the character that separates columns within each row (line) of the file. The default is a tab character in text format, a comma in CSV format. This must be a single one-byte character. This option is not allowed when using binary format.

Quote
Specifies the quoting character to be used when a data value is quoted. The default is double-quote. This must be a single one-byte character. This option is allowed only when using CSV format.

Escape
Specifies the character that should appear before a data character that

- FIC_CATEGORIE.csv

Import/Export data - table 'categorie'

General Options Columns

Import/Export

Import

Filename C:\Users\HP\Desktop\sgbd\mini projet\FIC_CATEGORIE.csv

Format csv

Encoding WIN1252

Close Reset OK

Import/Export data - table 'categorie'

General Options Columns

OID ☐

Header ☒

Delimiter ;

Quote "

Escape '.

Close Reset OK

- FIC_SOUS_CATEGORIE

Import/Export data - table 'sous_categorie'

General Options Columns

Import/Export: Import

Filename: C:\Users\HP\Desktop\sgbd\mini projet\FIC_SOUS_CATEGORIE.csv

Format: csv

Encoding: WIN1252

Close Reset OK

Import/Export data - table 'sous_categorie'

General Options Columns

OID: ☐

Header: ☒

Delimiter: ;

Quote: "

Escape: '

Close Reset OK

- FIC_ECRITURE.csv

Pour la table ecriture , on a utilisé le script suivant pour faire l'import :

```
\copy etudiants."ecriture" to
'C:\Users\HP\Desktop\sgbd\mini projet\FIC_ECRITURE.csv'
DELIMITER ';' CSV HEADER;
```

```
pise=# \copy etudiants."ecriture" to 'C:\Users\HP\Desktop\sgbd\mini projet\FIC_ECRITURE.csv' DELIMITER ';' CSV HEADER;
COPY 2264
pise=#
```

- FIC_MODE_REGLEMENT.csv

Import/Export data - table 'mode_reglement'

General Options Columns

Import/Export

Filename C:\Users\HP\Desktop\sgbd\mini projet\FIC_MODE_REGLEMENT.csv

Format csv

Encoding WIN1252

Close Reset OK

Import/Export data - table 'mode_reglement'

General Options Columns

OID ☐

Header ☒

Delimiter ;

Quote "

Escape '.

Close Reset OK