TD N°7 (Correction) : logiciel de comptabilité

Thème : modélisation d'un logiciel de saisie comptable

Contexte

Le service informatique de la société d'assurances ASSUTOUT est actuellement chargé de développer en interne un logiciel destiné à remplacer le progiciel de comptabilité que le service comptable utilise actuellement. Les futurs utilisateurs du logiciel ont décidé de le surnommer Compta+. Ce choix stratégique a été effectué par la DSI, en accord avec la direction comptable et les autres divisions de l'entreprise, afin de palier à terme certaines problématiques rencontrées avec le précédent progiciel.

En particulier, grâce à ce nouveau logiciel, la DSI espère pouvoir parfaitement interconnecter son SI métier au logiciel de comptabilité. Ainsi, la direction espère obtenir des gains de temps grâce à la mise en place de passerelles avec Compta+ et grâce à l'automatisation de certains traitements. Elle espère qu'il en résultera également de nombreuses économies : coût de formation, coût de maintenance, coût de développement de la part de l'éditeur, licences d'utilisation, etc.

Cependant, afin de disposer de ce logiciel dans un délai bref, il a été convenu de réaliser une application dont les fonctionnalités seront limitées. Ainsi, dans un premier temps, le logiciel permettra essentiellement la saisie et le suivi comptable. Dans un second temps, l'on envisagera de faire évoluer le logiciel et d'y adjoindre de nouvelles fonctionnalités.

Afin d'assurer cette transition, ASSUTOUT a décidé, de façon plutôt originale d'externaliser temporairement une partie de sa comptabilité auprès d'un cabinet d'expertise comptable dès lors que Compta+ sera mis en service. En effet, ASSUTOUT sous-traitera momentanément la production des documents comptables annuels, encore appelés « liasse fiscale », du fait que Compta+ n'automatisera pas initialement la production des documents en question.

D'un point de vue technologique, l'application sera développée :

- Pour la partie serveur : à l'aide d'une couche de web services développée en Java/JEE à l'aide du framework Spring. La partie serveur comportera une couche métier, mise en œuvre grâce à l'ORM hybernate. Cette couche métier et les web services seront interconnectés à une base de données relationnelles MySQL.
- Pour la partie client : à l'aide d'un front office AngularJS, HTML5 et CSS3.

Dans ce contexte, ASSUTOUT fait appel à vos services, à titre de *database engineer*, afin d'implémenter la base de données et les requêtes nécessaires au logiciel.

Travail à faire

1. Modélisation

A l'aide de l'annexe 1, répondre à la question suivante.

1.1.	Modéliser la base de données du logiciel sous forme de MCD (ou de diagramme de classes) au regard des informations fournies.
1.2.	Rédiger le schéma relationnel correspondant à votre MCD.

1.1. MCD : ce qu'il fallait voir.

- Une contrainte d'inclusion entre les associations « Pièce de référence » et « Pièces de l'écriture »
- Une écriture comptable est relative à un exercice : identifiant relatif (1,1) ou 1,1 (R)
- Une ligne d'écriture comptable est relative à une écriture comptable.

1.2. Schéma relationnel :

Exercice(Debut, Fin)
Clef primaire : Debut

Journal(Code, Libelle)
Clef primaire : Code

Compte(Num, Libelle)
Clef primaire : Num

Piece(Num, URI)
Clef primaire : Num

Ecriture(Exercice, Num, Journal, Piece, Creation, Libelle, Valide)
Clef primaire : Exercice, Num
Clefs étrangères :
- Exercice en référence à Exercice(Debut)
- Journal en référence à Journal(Code)
- (Exercice, Num, Piece) en référence à EcriturePiece(Exercice, Ecriture, Piece)
N.B. :
- Creation représente la date de passation de l'écriture comptable
- Piece correspond à la pièce de référence de l'écriture comptable
- Valide indique si l'écriture a été validée ou si elle ne l'a toujours pas été

EcriturePiece(Exercice, Ecriture, Piece)
Clef primaire : Exercice, Ecriture*, Piece*
Clefs étrangères :
- (Exercice, Ecriture*) en référence à Ecriture(Exercice, Num)*
- Piece en référence à Piece(Num)

EcritureLigne(Exercice, Ecriture, Num, Compte, Debit, Credit)

Clef primaire : Exercice, Ecriture, Num
Clefs étrangères :
- (Exercice, Ecriture) en référence à Ecriture(Exercice, Num)
- Compte en référence à Compte(Num)

2. Identification des contraintes d'intégrité

La société ASSUTOUT souhaite disposer d'un logiciel fiable et robuste. Dans ce contexte, elle vous a chargé d'élaborer une base dont les données doivent rester intègres.

2.1.	Quelle est le sens de la contrainte d'inclusion figurant normalement sur votre diagramme ?
2.2.	Quelles sont les opérations qu'il faudra effectuer afin que cette contrainte soit vérifiée ?
2.3.	Certaines contraintes d'intégrité pesant sur la base de données n'ont pu être représentées au travers des questions 1.1 et 1.2. Identifier ces contraintes et en expliquer le sens.
2.4.	Décrire les opérations à effectuer afin de vérifier les contraintes citées en 2.3.

2.1. Sens de la contrainte d'inclusion

La contrainte d'inclusion figurant entre les associations « Pièce de référence » et « Pièces de l'écriture » signifie que, pour qu'une pièce soit « pièce de référence », il faut déjà qu'elle fasse partie des pièces de l'écriture. Autrement dit : être pièce de référence d'une écriture implique être une pièce de l'écriture.

2.2. Vérifications à effectuer pour s'assurer que la contrainte d'inclusion soit vérifiée

On pourrait mettre en place un trigger qui, lors de l'insertion ou de la modification d'une écriture, vérifierait que la pièce de référence spécifiée (triplet Ecriture.Exercice, Ecriture.Num, Ecriture.Piece) figure bien dans la table EcriturePiece.
Cependant, une simple clef étrangère (voir schéma relationnel ci-avant) suffit ici à valider la contrainte d'inclusion. En effet, la clef étrangère nous assure que la pièce de référence (triplet cité ci-dessus) fasse bien référence à une pièce de EcriturePiece. Hormis la mise en place de cette, rien à faire…

2.3. Contraintes ne figurant ni sur le MCD ni sur le schéma relationnel :

En ce qui concerne les écritures et leurs lignes :
(a) Les champs EcritureLigne(Debit) et EcritureLigne(Credit) doivent être positifs ;
(b) Une fois une écriture validée, ses lignes ne sont plus modifiables (elles deviennent « stable ») ;
(c) La date de passation de l'écriture, à savoir la valeur du champ Ecriture(Creation), doit être comprise entre les dates de début et de fin de l'exercice auquel l'écriture est attachée.

En ce qui concerne les exercices, ils se suivent, et on a les contraintes suivantes :
(d) Date de début d'exercice <= Date de fin d'exercice ;
(e) Pas de chevauchement : les dates de début et de fin d'un exercice ne peuvent être comprise entre les dates de début et de fin d'un autre exercice.

2.4. Opération à effectuer pour vérifier les contraintes citées ci-avant :

(a) Vérifier que Debit > 0 et Crédit > 0 grâce à deux contraintes de domaine sur EcritureLigne : CHECK Debit >=0 et CHECK Credit >= 0.
(b) Les lignes d'une écriture peuvent être modifiées l'occasion d'un INSERT, d'un UPDATE ou d'un DELETE sur table EcritureLigne. A ces moments, grâce à un trigger, il convient de rejeter l'insertion, la modification ou la suppression si la condition Ecriture.Valide = TRUE est vérifiée pour l'écriture à laquelle la ligne d'écriture est associée.
(c) Grâce à un trigger, lors de l'insertion ou de la modification d'une écriture, il convient de vérifier que la date de l'écriture soit comprise entre les dates de début et de fin de l'exercice auquel elle est attachée. Si la vérification échoue, on rejette l'insertion ou la modification.
(d) Il convient d'ajouter la contrainte de domaine CHECK Debut <= Fin sur la table Exercice.
(e) Il convient de mettre un trigger en place lors de l'insertion et la modification d'un exercice : l'insertion ou la modification de l'exercice. Concernant l'insertion, elle doit échouer s'il existe un exercice E tel que : NEW.Debut **BETWEEN** E.Debut **AND** E.FIN **OR** NEW.FIN **BETWEEN** E.Debut **AND** E.FIN.

3. Requêtage

Afin de faciliter la conception des web services du logiciel, il vous est demandé de rédiger un certain nombre de requêtes qui pourront être utiles au programmeur qui s'en chargera.

3.1.	Rédiger la requête qui permet d'afficher la liste des écritures comptables du mois en cours, libellé inclus, avec chacune des lignes de l'écriture comptable. *N.B. : les fonctions CURDATE() et NOW() permettent de récupérer la date courante respectivement sous la forme AAAA-MM-JJ e AAAA-MM-JJ hh:mm:ss.*
3.2.	Rédiger la requête qui permet d'afficher la liste des écritures comptable de l'exercice en cours, libellé inclus, avec chacune des lignes de chaque écriture comptable. On souhaite également visualiser la référence et l'URI de la pièce comptable de référence de l'écriture comptable.
3.3.	Créer les vues « ecritures_exercices » et « ecritures_mois » permettant d'afficher le résultat des requêtes 3.1 et 3.2. *N.B. : afin de ne pas réécrire les deux requêtes précédentes, on abrègera par : (3.1) et (3.2).*
3.4.	Rédiger une adaptation de la requête 3.2 afin de n'afficher que les écritures comptables rattachées au journal dont le code est « BANQ ».

3.1. Liste des écritures comptables du mois en cours :

```
SELECT Ecriture.*, EcritureLigne.*, Compte.Libelle, Piece.URI
FROM Ecriture E
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
INNER JOIN Compte C ON L.Compte = C.Num
INNER JOIN Piece P ON E.Exercice = P.Exercice AND E.Num = P.Ecriture AND E.Piece = P.Num
WHERE MONTH(E.Creation) = MONTH(CURDATE())
AND YEAR (E.Creation) = YEAR(CURDATE())
```

3.2. Liste des écritures comptables de l'exercice en cours :

```
SELECT Ecriture.*, EcritureLigne.*, Compte.Libelle, Piece.URI
FROM Exercice EX
INNER JOIN Ecriture E ON EX.Debut = E.Exercice
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
INNER JOIN Compte C ON L.Compte = C.Num
INNER JOIN Piece P ON E.Exercice = P.Exercice AND E.Num = P.Ecriture AND E.Piece = P.Num
WHERE CURDATE() BETWEEN EX.Debut AND EX.Fin
```

3.3. Deux vues :

```
CREATE VIEW ecritures_mois AS {requête 3.1.}
CREATE VIEW ecritures_exercice AS {requête 3.2.}
```

3.4. Liste des écritures comptables de l'exercice en cours et inscrites dans le journal « BANQ » :

```
SELECT Ecriture.*, EcritureLigne.*, Compte.Libelle, Piece.URI
FROM Exercice EX
INNER JOIN Ecriture E ON EX.Debut = E.Exercice
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
INNER JOIN Compte C ON L.Compte = C.Num
INNER JOIN Piece P ON E.Exercice = P.Exercice AND E.Num = P.Ecriture AND E.Piece = P.Num
WHERE CURDATE() BETWEEN EX.Debut AND EX.Fin
AND E.Journal = 'BANQ'
```

Une écriture comptable est dite « équilibrée » si la somme de ses débits est égale à la somme de ses crédits.

3.5.	Rédiger la requête permettant d'afficher les écritures comptables avec, pour chaque écriture comptable, la somme de ses débits et crédits.
3.6.	Compléter la requête afin de préciser également si l'écriture est équilibrée. *N.B. : la fonction IIF(condition, valeurSiVrai, valeurSiFaux) permet d'afficher une valeur si une condition est vérifiée, une autre dans le cas contraire.*
3.7.	Rédiger la requête permettant d'afficher la somme de débits et crédits d'un exercice comptable.

3.5. Les écritures comptables de l'exercice en cours avec la somme de leurs débits et crédits :

```
SELECT E.Num, E.Libelle, SUM(L.Debit) AS 'Débits', SUM(L.Credit) AS 'Credits'
FROM Exercice EX
INNER JOIN Ecriture E ON EX.Debut = E.Exercice
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
WHERE CURDATE() BETWEEN EX.Debut AND EX.Fin
GROUP BY E.Num, E.Libelle ;
```

3.6. Les écritures comptables de l'exercice en cours avec la somme de leurs débits et de leurs crédits ainsi que l'indication « Equilibré » ou « Non équilibré » selon li l'écriture est ou n'est pas équilibrée :

```
SELECT
    E.Num, E.Libelle,
    SUM(L.Debit) AS 'Débits', SUM(L.Credit) AS 'Credits',
    IF(SUM(L.Debit) = SUM(L.Credit), 'Equilibré', 'Non équilibré')
FROM Exercice EX
INNER JOIN Ecriture E ON EX.Debut = E.Exercice
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
WHERE CURDATE() BETWEEN EX.Debut AND EX.Fin
GROUP BY E.Num, E.Libelle ;
```

3.7. Sommes des débits et crédits par exercice comptable :

```
SELECT EX.Debut, EX.Fin, SUM(L.Debit) AS 'Débits', SUM(L.Credit) AS 'Credits'
FROM Exercice EX
INNER JOIN Ecriture E ON EX.Debut = E.Exercice
INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
GROUP BY EX.Debut, EX.Fin ;
```

En comptabilité :

- Résultat = Produits - Charges
- Les produits sont répertoriés dans les comptes de classe 7, c'est-à-dire les comptes commençant par la « lettre » 7. Les charges sont répertoriées quant à elles dans ceux commençant par la « lettre » 6.
- Produits = Crédits comptes 7… - Débits comptes 7… (on dit que les produits augmentent au crédit)
- Charges = Débits compte 6… - Crédits comptes 6… (on dit que les charges augmentent au débit).

3.8.	Rédiger la requête permettant de calculer le résultat de l'exercice comptable en cours.

3.8. Le résultat de l'exercice en cours:

```
SELECT P.Exercice, P.Produits, C.Charges, P.Produits - C.Charges AS 'Résultat'
FROM (
   SELECT L.Exercice AS 'Exercice', SUM(L.Debit) - SUM(L.Credit) AS 'Produits'
   FROM EcritureLigne L
   WHERE L.Compte LIKE '7%'
   GROUP BY L.Exercice
) P INNER JOIN (
   SELECT L.Exercice AS 'Exercice', SUM(L.Debit) - SUM(L.Credit) AS 'Charges
   FROM EcritureLigne L
   WHERE L.Compte LIKE '6%'
   GROUP BY L.Exercice
) C ON P.Exercice = C.Exercice
INNER JOIN Exercice E ON C.Exercice = E.Debut
WHERE CURDATE() BETWEEN E.Debut AND E.FIN ;
```

4. Contrôle de l'intégrité des données

4.1.	Donner le code du trigger permettant de vérifier la contrainte d'inclusion évoquées en 2.1 et dont les opérations ont été décrites en 2.2.
4.2.	Donner le code du ou des triggers implémentant le contrôle des contraintes successivement évoquées puis décrites en 2.3 et 2.4.
4.3.	Rédiger le code du trigger qui empêche de valider une écriture comptable non équilibrée.

4.1. Code du trigger permettant de vérifier la contrainte d'inclusion évoquée en 2.1 et 2.2.

Aucun trigger n'est nécessaire afin de vérifier cette contrainte d'inclusion (voir 2.2). Si vous souhaitez le rédiger, à votre guise. Votre trigger fera en l'occurrence le « job » d'une vérification de clef étrangère.

4.2. Vérification des contraintes identifiées en 2.3 et 2.4

(a) Vérifier que Debit > 0 et Crédit > 0 grâce à deux contraintes de domaine sur EcritureLigne : CHECK Debit >=0 et CHECK Credit >= 0.

```
ALTER TABLE EcritureLigne ADD CONSTRAINT dc_positif CHECK (Debit >= 0 AND Credit >= 0) ;
```

(b) Les lignes d'une écriture peuvent être modifiées l'occasion d'un INSERT, d'un UPDATE ou d'un DELETE sur table EcritureLigne. A ces moments, grâce à un trigger, il convient de rejeter l'insertion, la modification ou la suppression si la condition Ecriture.Valide = TRUE est vérifiée pour l'écriture à laquelle la ligne d'écriture est associée.

```
DROP TRIGGER IF EXISTS ecriture_stable;
DELIMITER $$
CREATE TRIGGER ecriture_stable BEFORE UPDATE, DELETE ON ecriture
BEGIN
    IF old.valide = TRUE THEN
        SIGNAL sqlstate '45000'
            SET message_text = 'L'écriture a été validée, impossible de modifier';
    END IF;
END $$
DELIMITER ;

CREATE TRIGGER ecriture_ligne_stable BEFORE INSERT, UPDATE ON ecriture_ligne
BEGIN
    SELECT valide INTO @valide
    FROM Ecriture
    WHERE Ecriture.Exercice = NEW.Exercice
    AND Ecriture.Num = NEW Ecriture ;
    IF @valide THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Ecriture validée ! Modif. Interdite !'
    END IF ;
END
```

N.B. : on doit placer un trigger similaire BEFORE DELETE. Seul changement : OLD au lieu de NEW.

(c) Grâce à un trigger, lors de l'insertion ou de la modification d'une écriture, il convient de vérifier que la date de l'écriture soit comprise entre les dates de début et de fin de l'exercice auquel elle est attachée. Si la vérification échoue, on rejette l'insertion ou la modification.

```
DELIMITER $$
CREATE TRIGGER verif_delete BEFORE INSERT,UPDATE ON Ecriture
BEGIN
        SELECT dateDebut, dateFin INTO @deb, @fin
        FROM exerciceComptable E
        WHERE E.debut = new.Exercice ;
        IF @deb>=new.creation || @fin<=new. Creation THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEST = 'Date d\'écriture invalide !' ;
        END IF ;
END $$
DELIMITER ;
```

(d) Il convient d'ajouter la contrainte de domaine CHECK Debut <= Fin sur la table Exercice.

```
ALTER TABLE Exercice ADD CHECK (Debut <= Fin) ; -- Le nom de  la contrainte est optionnel
```

N.B. : valable sous réserve que la comparaison de champ soit prise en charge par le SGBD.

(e) Il convient de mettre un trigger en place lors de l'insertion et la modification d'un exercice : l'insertion ou la modification de l'exercice. Concernant l'insertion, elle doit échouer s'il existe un exercice E tel que : NEW.Debut **BETWEEN** E.Debut **AND** E.FIN **OR** NEW.FIN **BETWEEN** E.Debut **AND** E.FIN.

```
CREATE TRIGGER verif_chevauchement  ON Exercice BEFORE INSERT, UPDATE FOR EACH ROW
BEGIN
    SELECT COUNT(*) INTO @Nb
    FROM Exercice E
    WHERE NEW.Debut BETWEEN E.Debut AND E.Fin OR NEW.Fub BETWEEN E.Debut AND E.Fin ;
    IF @Nb > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Chevauchement !  Dates de l\'exercice invalides !' ;
    END IF ;
END ;
```

4.3. Trigger empêchant la validation d'une écriture non équilibrée

```
CREATE TRIGGER valid_equlibre ON Ecriture BEFORE INSERT, UPDATE FOR EACH ROW
BEGIN
    -- En cas de validation d'écriture
    IF NEW.Valide = 1 THEN
        -- Récupération des débits et crédits de l'écriture à valider
        SELECT SUM(Debit), SUM(Credit) INTO @TotalDebits, @TotalCredits
        FROM Ligne L
        WHERE L.Exercice = NEW.Exercice
        AND L.Ecriture = NEW.Num ;
        IF @TotalDebits <> @TotalCredits THEN -- Si écriture non équilibrée
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Ecriture non équilibrée ! Echec validation !' ;
        END IF ;
    END IF ;
END ;
```

5. Utilitaires

Afin une fois encore de faciliter la conception des web services de l'application, il vous est demandé de développer un certain nombre de procédures stockées auquel le programmeur des web services pourra faire appel.

5.1.	En vous inspirant de la réponse apportée à la question 3.4., rédiger la procédure stockée « ecritures_journal » permettant d'afficher les écritures comptables (libellé, lignes, pièce de référence, etc.) relative à un journal dont le code est passé en paramètre.
5.2.	En vous inspirant de la réponse à la question 3.8., rédiger « exercice_resultat », la procédure stockée retournant en paramètre de sortie le résultat de l'exercice en cours.

5.1. Affichage des écritures d'un journal donné

```
CREATE PROCEDURE ecritures_journal(unCode VARCHAR)
BEGIN
   -- Reprise de la requête 2.4.
   SELECT Ecriture.*, EcritureLigne.*, Compte.Libelle, Piece.URI
   FROM Exercice EX
   INNER JOIN Ecriture E ON EX.Debut = E.Exercice
   INNER JOIN EcritureLigne L ON E.Exercice = L.Exercice AND E.Num = L.Ecriture
   INNER JOIN Compte C ON L.Compte = C.Num
   INNER JOIN Piece P ON E.Exercice = P.Exercice AND E.Num = P.Ecriture AND E.Piece = P.Num
   WHERE CURDATE() BETWEEN EX.Debut AND EX.Fin
   AND E.Journal = unCode ; -- ligne modifiée
END ;
```

5.2. Affichage du résultat de l'exercice en cours

```
CREATE PROCEDURE exercice_resultat()
BEGIN
   {requête 3.8.}
END ;
```

Annexe 1 : informations transmises par la responsable du service comptabilité

Grégory,

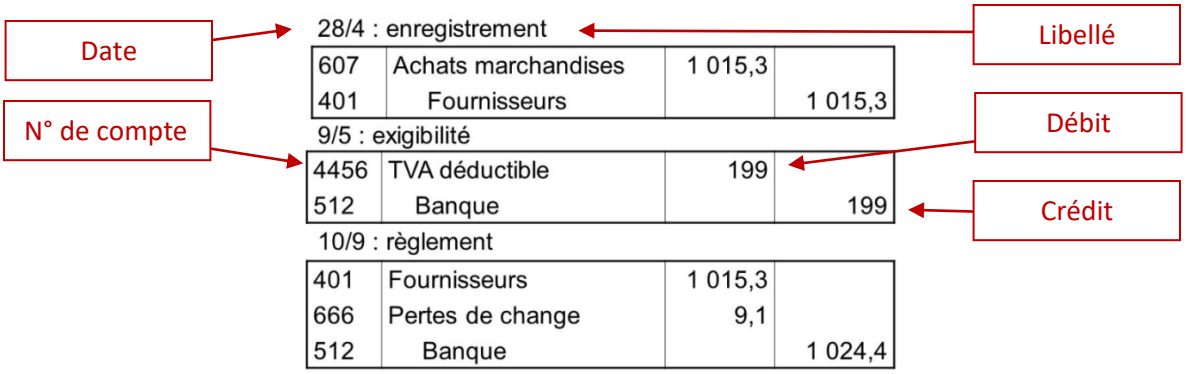
[…]

Pour faire suite à notre dernière entrevue, et comme nous en avons convenu, voici une explication de la logique comptable et plus particulièrement de la saisie comptable. Tu trouveras également ci-après un exemple d'écriture comptable.

1. La logique comptable

- Tout d'abord, une comptabilité porte sur un exercice comptable ayant une date de début et une date de fin ;
- Les exercices comptables se suivent. Par exemple, notre exercice comptable actuel est celui allant du 01/01/2017 au 31/12/2017. L'exercice suivant sera donc celui allant du 01/01/2018 au 31/12/2018 ;
- Par ailleurs, le métier de comptable consiste en outre à saisir ce qu'on appelle des écritures comptables. Une écriture comptable est toujours datée. Et on lui donne un libellé.
- Une écriture est bien entendue reliée à un exercice ;
- La comptabilité est divisée en ce qu'on appelle des journaux. Globalement, un journal a surtout un code journal (exemple : « BANQ », « PAIE », « EXTO », etc.) et une libellé. Cela permet de classer les écritures. En effet, une écriture est toujours rattachée à un journal ;
- Sinon, on peut en général attacher des pièces comptables (=documents) à une écriture. Parmi ces pièces, l'écriture finira toujours par se voir attribuer une pièce de référence. Bien entendu, il faudra pouvoir consulter la pièce avec un lien ou un truc du genre ;
- Au demeurant, une écriture est constituée de lignes. Chaque ligne de l'écriture peut avoir un montant (€) au débit et/ou au crédit. Une ligne d'écriture est attachée à un compte du plan de comptes ;
- Le plan de comptes, c'est tous les comptes de l'entreprise (exemple : 512 - Banque) ;
- Ah oui ! Une fois qu'on a validé une écriture comptable, il ne faut surtout plus pouvoir la modifier ni modifier ses lignes.

2. Un exemple d'écriture



[…]