Julien Lefevre
Antoine Promis
Arnaud Bonnet
Téo Itsweire-Krebs

L3 Informatique
3 Mai 2024
Base de Données 2

Construction et exploitation d'une base de données pour une plateforme de vidéo à la demande (VOD)

Table des matières

Introduction

Dans le cadre de notre Troisième année de Licence Informatique à l'Université de Poitiers, l'Unité d'Enseignement Base De Données 2 nous propose la réalisation d'un projet ayant pour objectif la conception et l'exploitation d'une base de données pour la plateforme de vidéo à la demande *CanalPlusRiche*.

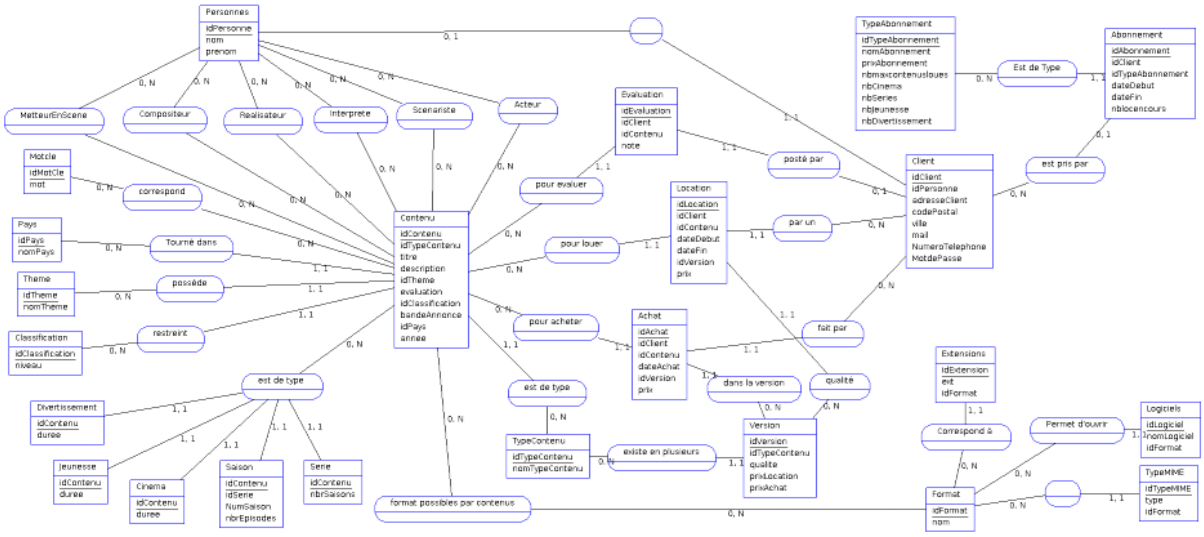
Dans un premier temps, nous présenterons les livrables demandés, puis nous parlerons de la répartition du travail dans notre groupe.

Livrable n°1

Présentation de la base de données et du MCD

Le MCD a été réalisé avec le logiciel AnalyseSI.

Nous avons décidé d'organiser notre base de données autour des contenus, il s'agit de l'élément central du sujet et de la base de toute plateforme de VOD. Nous expliquerons en détail le schéma en dessous. Il y a 2 grandes parties qui sont obligatoires pour une plateforme de VOD, la première correspond à la gestion des contenus et des informations qui vont avec (format, type, thème, mots clés ...), la seconde partie correspond à la gestion des clients et leurs actions (abonnement, location, achat et évaluation des contenus).



Les Contenus et les tables associées :

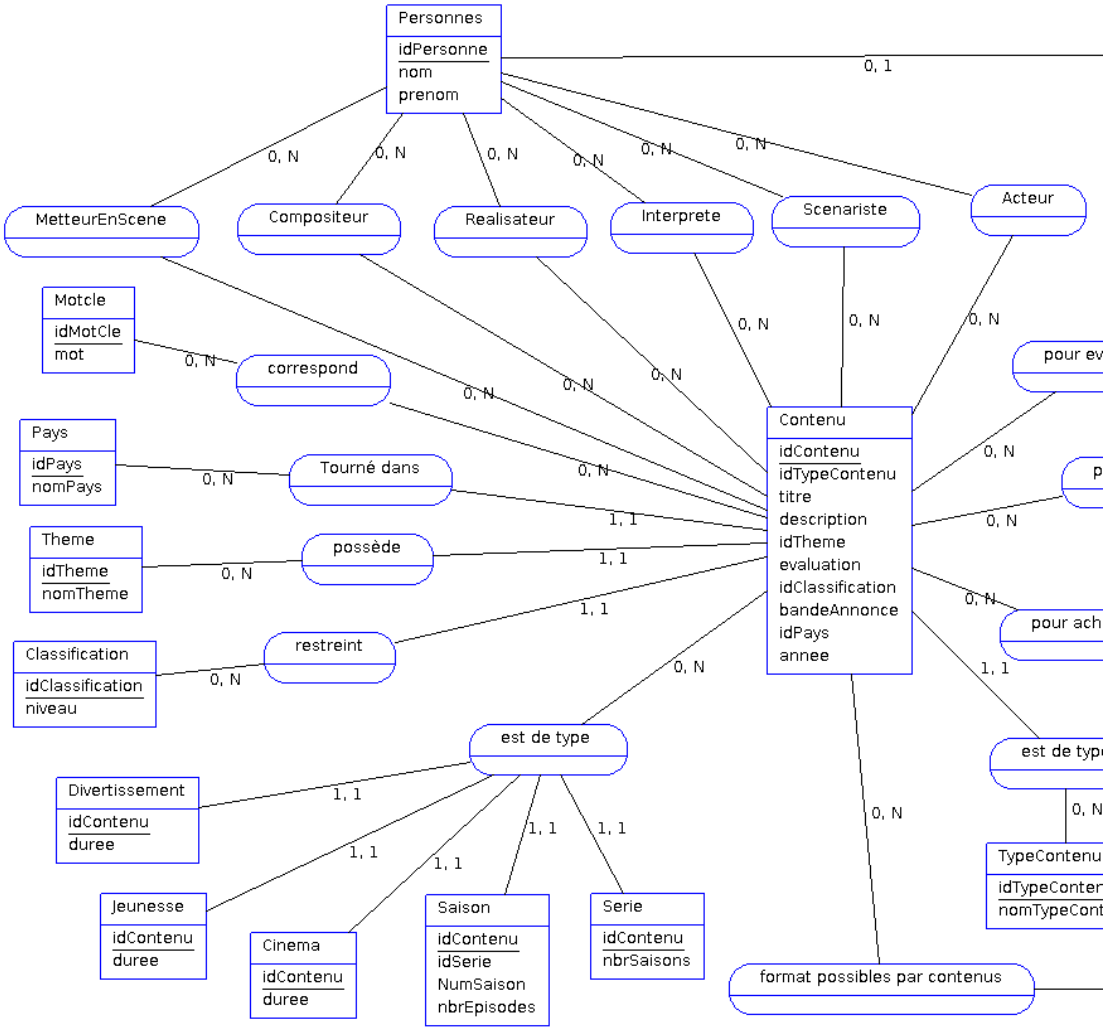
Les contenus sont gérés grâce à un identifiant unique qui est la clé primaire de la table Contenu. Cette table contient les informations communes à tous les contenus comme le titre, le thème ou encore la description. On peut remarquer les associations avec les tables Pays, Thème et Classification. Ces associations permettent d'avoir un seul élément de chaque table pour un contenu donné mais permettent à l'inverse d'avoir plusieurs contenus avec le même pays par exemple. Ces informations sont donc stockées grâce à des clés étrangères dans la table Contenu.

Les mots clés sont gérés légèrement différemment, il s'agit d'une relation O-N vers O-N qui permet de limiter le nombre de mots clés stockés et également d'en mettre plusieurs par contenus, ce qui est généralement le cas sur un site de VOD. Grâce à cette

association, nous sommes sûr qu'un mot clé utilisé pour plusieurs contenus sera le même et cela permet si nous le voulons de rechercher des films par mot clé.

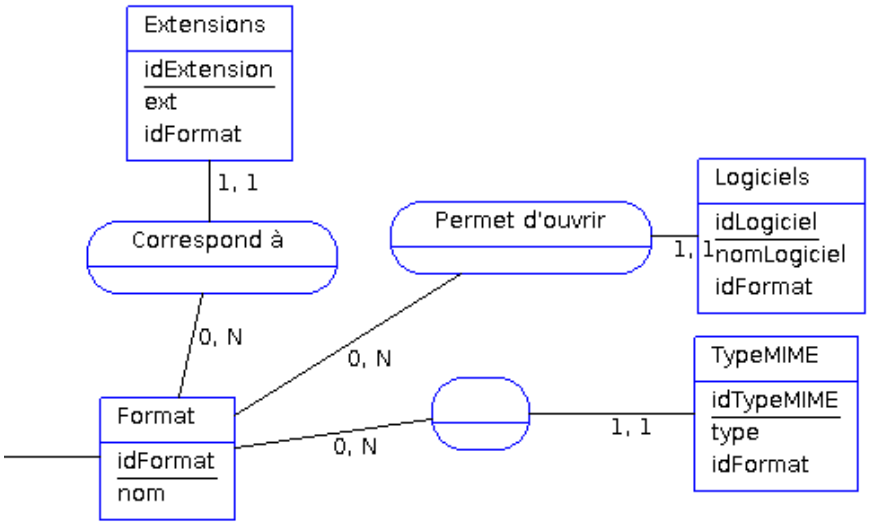
Pour les types de contenus nous les stockons sur une table à part qui permet de limiter aux 5 types spécifiés dans le sujet (Cinéma, Divertissement, Séries, Saisons, Jeunesse). Ces types correspondent à des tables qui nous permettent de gérer des informations spécifiques à chaque type.

Enfin nous avons décidé de créer une table nommée Personnes qui stocke toutes les personnes de la base de données qu'elles soient des clients ou non. Nous avons des relations plusieurs à plusieurs entre la table Personnes et la table Contenu qui nous permettent de préciser les acteurs, réalisateurs ou autres membres ayant participé à la création d'un contenu.



Les formats sont liés aux contenus avec une association plusieurs à plusieurs qui permet plusieurs formats pour les contenus. Chaque format possède une ou plusieurs

extensions, peut être ouvert par un ou plusieurs logiciels et a un ou plusieurs types MIME.



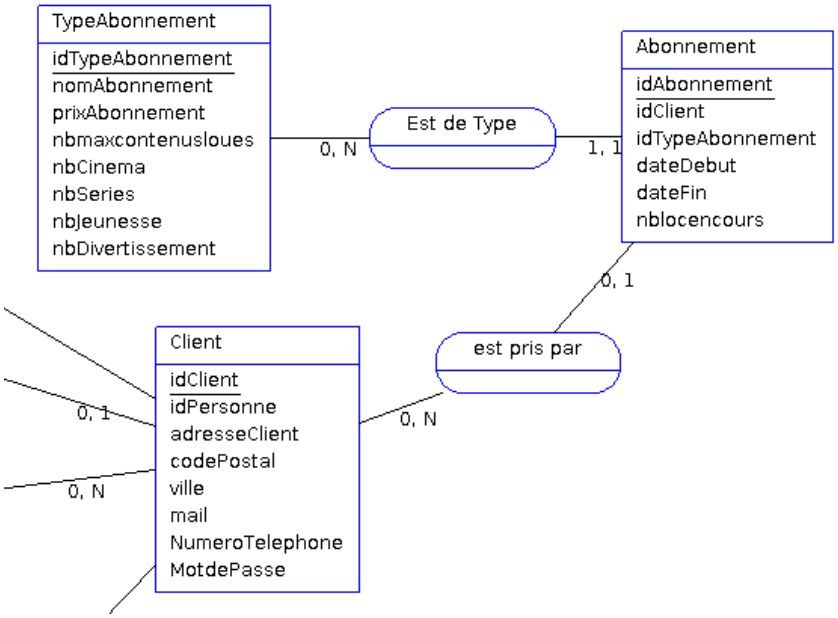
Les clients et les abonnements :

Les clients sont des personnes et sont donc logiquement reliés à la table Personnes. La table Client contient donc une clé étrangère qui référence une personne cela nous permet d'éviter d'avoir un client pour plusieurs personnes. Mais cela nous permet également d'avoir un acteur qui serait aussi un client par exemple sans avoir à dupliquer les données le concernant comme le prénom ou le nom de famille.

Un client peut souscrire à un abonnement s'il le souhaite, mais il est obligé de créer un compte pour s'abonner ou même acheter ou louer un contenu.

Nous gardons l'identifiant du client dans la table Abonnement en tant que clé étrangère. Cela permet de garder un historique des abonnements résiliés ou non pour toute la plateforme en sachant quel client a souscris l'abonnement. Cette façon de lier les clients aux abonnements nous évite également de se retrouver avec des champs potentiellement nuls si nous avions fait la relation dans le sens inverse. En effet, un client qui n'aurait pas souscris d'abonnement aurait eu un identifiant d'abonnement à NULL.

Enfin nous avons une table TypeAbonnement qui garde les informations spécifiques à chaque type d'abonnement (VIP, Premium, Basique) comme le prix par mois ou encore la durée de location selon le type de contenu. Il s'agit d'informations qui évoluent en fonction du type d'abonnement et qui doivent être modifiables facilement. Par exemple si nous décidons de mettre à jour les fonctionnalités d'un abonnement ou si nous souhaitons ajouter un nouveau type.



Les achats/locations/évaluations et les différentes versions disponibles

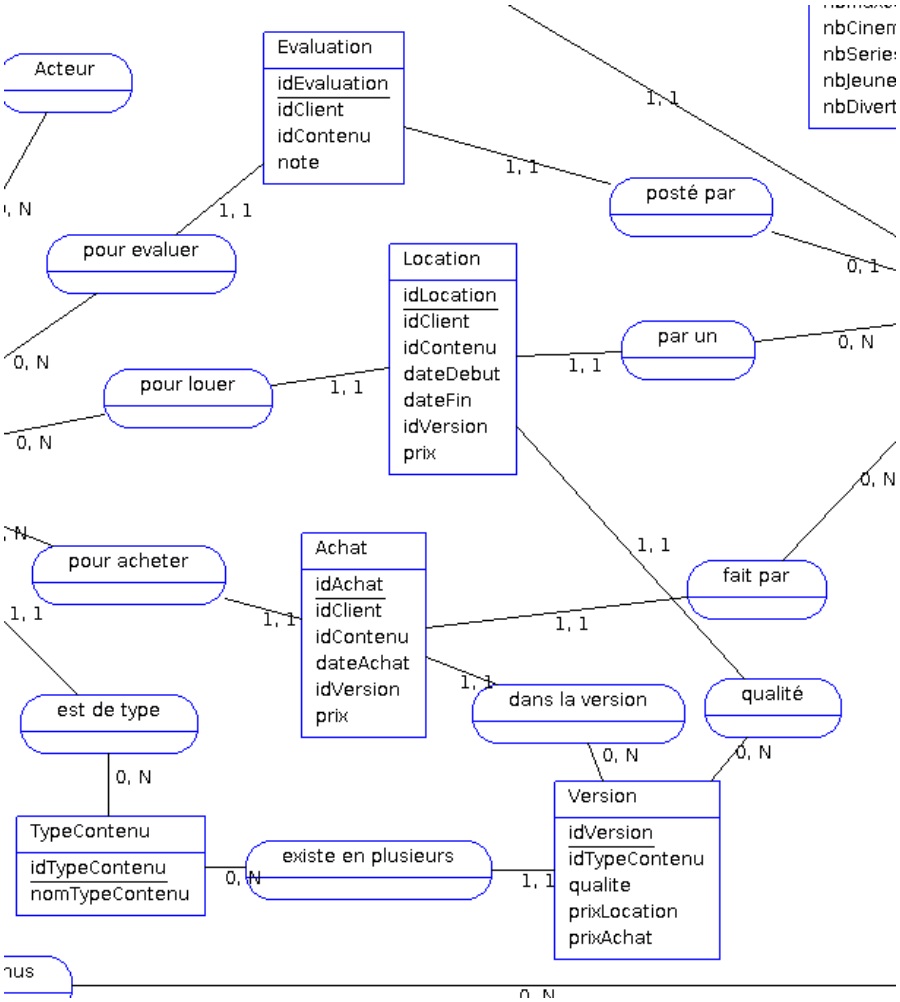
Un contenu peut être acheté ou loué, il s'agit du même mécanisme avec une légère différence pour la location qui possède une date de fin contrairement à un achat.

Un achat/location est réalisé par un client, cette opération concerne un contenu, possède une date d'achat ou de début de location et a un prix en fonction de la version du contenu choisi. Cette version est pour l'instant soit HD soit SD. Le sujet ne précisant pas si le prix devait être propre à chaque contenu, nous avons décidé que chaque version correspond à un type de contenu et non à un contenu. Cela nous permet de faire des prix identiques pour tous les films, toutes les séries, chose qui est plus facile à gérer et qui nécessite beaucoup moins d'espace de stockage. Il y a donc un prix d'achat et un prix de location différent par versions et par type de contenu.

Enfin un client peut évaluer un contenu à la suite d'un achat ou d'une location. L'évaluation concerne un contenu et un client, il ne peut pas y avoir plusieurs évaluations du contenu pour un même client pour empêcher de fausser la note moyenne du contenu.

Pour être sûr qu'un client puisse noter seulement les contenus qu'il a acheté ou loué nous avons mis en place un déclencheur qui vérifie s'il existe une ligne dans la table location ou la table achat correspondant au couple identifiant du client et identifiant du contenu. Si c'est le cas, l'évaluation peut être faite, sinon elle est annulée et ne sera pas insérée dans la table.

Les tables Location, Achat et Evaluation fonctionnent sur le même principe que la table Abonnement. En effet, elles possèdent une clé étrangère faisant référence à un client afin de garder un historique des achats et des locations de chaque client associé au contenu qui a été visionné.



Livrable n°2

La création des tables

Pour ce livrable, que vous pourrez retrouver sous le nom ***create_tables.sql*** , nous avons retranscris en SQL notre MCD, mais nous y avons aussi ajouté des séquences et déclencheurs permettant l'auto incrémentation des identifiants de certaines tables. Aussi, vous retrouverez à la fin de ce fichier 3 déclencheurs qui gèrent certaines contraintes spécifiques. Par exemple, à l'ajout d'une Location, on met à jour le nombre de contenus en cours de location (si le client a un abonnement) et on détermine la date de fin de la location selon la situation du client. Les deux autres déclencheurs s'occupent de la gestion des évaluations, à savoir lors de l'insertion d'une évaluation, on vérifie que le client avait bien acheté ou loué le contenu qu'il évalue (1er déclencheur) puis on met à jour la note évaluation du contenu (2nd déclencheur).

Choix de conception

Conventions d'écritures

Nous avons mis en place quelques conventions d'écriture pour faciliter la lecture et la compréhension des éléments de la base de données.

Les clés primaires sont nommées sous la forme : PK_NOMTABLE, cela permet donc de savoir qu'il s'agit de la clé primaire (PK) de la table (NOMTABLE).

Pour les clés étrangères nous avons choisi la forme : FK_NOMTABLE_NOMTABLEETRANGERE qui nous permet de connaître en premier la table de la clé étrangère et en deuxième la table référencée, on commence par FK pour "foreign key".

Livrable n°3

L'insertion dans notre base de données

La composition de ce livrable, que vous retrouverez sous le nom ***insert.sql*** , est un peu particulière, nous allons donc la décrire rapidement ici. Les 8000+ premières lignes sont occupées par les valeurs de *datanetflouze*, c'est après que vous retrouverez nos insertions.

Concernant les insertions, nous avons choisis de décortiquer les informations de la tables *datanetflouze* ainsi que des vues qui nous ont été transmises. Ainsi, nous avons créé de multiples vues enregistrant pour la plupart seulement une information relative à chaque contenu. Par exemple, la vue *v_rating* contient le *show_id* du contenu ainsi que sa classification selon le barème imposé (-12, Tous publics, etc). Ensuite nous avons fait des jointures entre toutes ces vues pour insérer les contenus.

Une autre spécificité de nos insertions réside dans la création d'une table temporaires *TempPersonnes*, dans laquelle nous récupérions tous les noms issus des vues *v_cast* et *v_director*, et ce afin d'éviter les doublons lors de l'insertion dans la table Personnes, car certains acteurs sont aussi réalisateurs.

Livrable n°4

L'insertion de clients

Ce fichier, que vous retrouverez sous le nom ***clients.sql***, est construit de la manière suivante : dans un premier temps, nous insérons dans la table Personnes différents futurs clients, ensuite, nous créons des clients en utilisant les *idPersonne* que nous venions d'insérer dans la table Personnes, troisièmement, nous donnons à certains clients des abonnements, puis nous avons inséré quelques achats et locations avant de créer certaines évaluations.

Livrable n°5

Spécification de transactions

Pour ce livrable nous devions spécifier certaines transactions données. Nous avons choisi de les réaliser d'une façon similaire aux exemples présents dans le cours CMn°4.

Création d'un compte par un client et prise d'un abonnement

Variables (nom N, prenom P, Adresse A, mail M, Telephone T, Mot de Passe MDP, type Abonnement TA)

Début :

Rechercher idPersonne dans Personnes pour N et P

Si idPersonne != null Alors

Insertion dans Personne (new_idPersonne, N, P) ;

Insertion dans Client (new_idClient, new_idPersonne, A, M , T, MDP) ;

Insertion dans Abonnement (new_idAbonnement, new_idClient ,TA, aujourd'hui) ;

Sinon

Insertion dans Client (new_idClient, idPersonne, A, M, T, MDP) ;

Insertion dans Abonnement (new_idAbonnement, new_idClient, TA, aujourd'hui) ;

Fin Si ;

Fin ;

Résiliation d'un abonnement par un client (ce qui met fin à toutes ses locations en cours)

Variables (Identifiant Client C, identifiant Abonnement A)

Début :

Vérifier existence de C ;

Vérifier existence de A ;

Vérifier C = idClient dans A ;

Rechercher LS dans Location où idClient = C et (datefin = null ou datefin > aujourd'hui) ;

Boucle (pour chaque L dans LS) :

Mettre à jour datefin = aujourd'hui pour la Location L ;

Fin Boucle ;

Mettre à jour datefin = aujourd'hui pour l'Abonnement A ;

Fin ;

Location d'un contenu par un abonné

Variables (Identifiant Abonnement A, Contenu C, idVersion V)

Début :

Vérifier existence de A ;

Vérifier existence de C ;

Lecture dans DF de Datefin dans A ;

Lecture dans TA de idTypeAbonnement dans Abonnement ;

Lecture dans NBMAX de nbmaxcontenusloues dans TypeAbonnement pour idTypeAbonnement = TA ;

Lecture dans NBLOC de nblocencours dans Abonnement ;

Lecture dans IDCLIENT de idClient dans Abonnement ;

Lecture dans P de prixLocation dans Version pour idVersion = V ;

Si (DF = null OU DF > aujourd'hui) Alors

Si (NBLOC < NBMAX) Alors

Insertion dans Location de (new_idLocation, IDCLIENT, C, aujourd'hui, null, V, P) ;

Fin Si ;

Fin Si ;

Fin ;

Suppression d’un contenu par un administrateur sur la plateforme (ce qui met fin à toutes les locations en cours de ce contenu)

Variables (Compte_Client C, idContenu CT)

Début :

Vérifier existence de C ;

Vérifier C est administrateur ;

Vérifier existence de CT ;

Rechercher LOCS dans Location pour idContenu = CT ;

Boucle (Pour chaque L dans LOCS)

Lecture dans DF de datefin dans L ;

Si (DF = null OU DF > aujourd’hui) Alors

Mettre à jour Datefin = aujourd’hui dans L ;

Fin si ;

Fin Boucle ;

Suppression dans Contenu de CT ;

Fin ;

Livrable n°6

Afin d'éviter des problèmes de concurrence, nous devons mettre en place des verrous, vous trouverez ci-dessous lesquels et pourquoi.

Pour la fonction de création d'un compte et la prise d'un abonnement, il peut être judicieux d'utiliser un verrou de type SERIALIZABLE sur la table Client et sur la table Abonnement afin d'éviter les problèmes de concurrence lors de la recherche et de l'insertion des données.

Pour la fonction de résiliation d'un abonnement, un verrou EXCLUSIVE sur la table Abonnement serait nécessaire pour éviter les problèmes de concurrence lors de la mise à jour de la date de fin de l'abonnement et des locations associées.

Pour la fonction de location d'un contenu par un abonné, un verrou EXCLUSIVE sur la table Abonnement serait nécessaire pour éviter les problèmes de concurrence lors de la vérification de l'existence de l'abonnement et de la mise à jour du nombre de locations en cours.

Enfin, pour la suppression d'un contenu par un administrateur, un verrou EXCLUSIVE sur la table Location serait nécessaire pour éviter les problèmes de concurrence lors de la recherche et de la mise à jour des locations en cours sur le contenu à supprimer.

Livrable n°7

Concernant le niveau d'isolation, nous recommandons d'utiliser le niveau d'isolation SERIALIZABLE pour ces transactions. Ce niveau assure la plus haute isolation en garantissant que les transactions s'exécutent de manière séquentielle, évitant ainsi les problèmes de concurrence. Bien que cela puisse entraîner une légère baisse des performances en raison du verrouillage plus strict, cela garantit l'intégrité des données et la cohérence des opérations. Les autres niveaux d'isolation comme READ COMMITTED ou REPEATABLE READ peuvent présenter des problèmes de concurrence, notamment des phénomènes de lecture sale, de lecture non répétable ou de lecture fantôme, ce qui peut compromettre la fiabilité des opérations dans un environnement transactionnel comme celui-ci.

Livrable n°8

Insertion de contraintes

Ce fichier, que vous retrouverez sous le nom ***contraintes.sql*** , contient des contraintes induites par les souhaits de *CanalPlusRiche* d'avoir des données cohérentes.

Un détail que nous avons remarqué cependant est qu'il ne faut pas exécuter ce fichier avant le fichier nommé ***insert.sql*** . En effet, l'une des contraintes indique qu'un contenu pour la Jeunesse ne peut être déconseillé aux moins de 16 ou moins de 18 ans. Or il semblerait que l'un des contenus livrés par *datanetflouze* soit dans ce cas de figure, empêchant alors l'insertion de tous les autres contenus de *datanetflouze* dû à cette erreur.

Livrable n°9

Insertions de requêtes

Ce fichier, que vous retrouverez sous le nom ***requete.sql***, contient les 20 requêtes demandées par le client. Par manque de temps, nous n'avons pas optimisé ces requêtes (Livrable n°10).

Organisation du travail

Planning

Notre planning sur ce projet se divise en deux parties, la réunion entre les membres du groupe enfin de débattre sur certaines parties du sujets comme par exemple le MCD, puis la partie programmation.

Répartition du travail

Tous les membres du groupe ont participés à niveau égal sur la conception du MCD de ce projet. La partie programmation est légèrement plus déséquilibrée et ce dû à la présence d'autres projets, cependant ce déséquilibre est faible, tous les membres du groupes ayant participé à la partie programmation de ce projet. En effet, nous avons travaillé autant que possible tout ensemble en même temps, en se répartissant les tâches. Par exemple, sur le fichier ***create_tables.sql***, nous nous sommes répartis les tables à programmer, les clés étrangères ne posant pas de problème grâce aux conventions d'écriture que nous nous sommes fixé.

Conclusion

Pour conclure, nous pouvons noter dans les points négatifs le fait de ne pas avoir entièrement fini le projet, le livrable n°10 sur l'optimisation des requêtes manquant à l'appel. Cependant, nous avons été en mesure d'identifier tous les problèmes que nous avons rencontré, et nous avons réussi à les éliminer un par un.

Ce projet nous a apporté beaucoup d'expérience, autant dans le domaine des bases de données que dans la gestion d'un projet.