Rennequinepolis

Dossier explicatif de la maintenance

DESART Adrien & GUISSART Julien - 2301

2016

1. Table des matières

[I. Introduction 2](#_Toc460097347)

[II. Contexte général 3](#_Toc460097348)

[1. Les différents sites 3](#_Toc460097349)

[2. Description des éléments de RQS 3](#_Toc460097350)

[2.1. SYS 3](#_Toc460097351)

[2.1.1. AccessControlList.sql 3](#_Toc460097352)

[2.1.2. CreateUsers.sql 4](#_Toc460097353)

[2.2. CB/CBB 5](#_Toc460097354)

[2.2.1. Modèle relationnel de données 5](#_Toc460097355)

[2.2.2. CreaCB\_ScriptLDD.sql 6](#_Toc460097356)

[2.2.3. Package AlimCB 6](#_Toc460097357)

[2.2.4. Package BackupCB 9](#_Toc460097358)

[2.2.5. Create\_Job.sql 10](#_Toc460097359)

[2.2.6. Trigger\_ReplicationCopie.sql 10](#_Toc460097360)

[2.2.7. Trigger\_ReplicationCotesAvis.sql 10](#_Toc460097361)

[2.2.8. Package RestoreCB 11](#_Toc460097362)

[2.2.9. Package CrashCB 11](#_Toc460097363)

[2.2.10. Package RechFilm 12](#_Toc460097364)

[2.2.11. Package EvalFilm 13](#_Toc460097365)

[2.2.12. Package AlimCC 14](#_Toc460097366)

[2.3. CC 14](#_Toc460097367)

[2.3.1. Modèle relationnel de données 14](#_Toc460097368)

[2.3.2. CreaCC.sql 15](#_Toc460097369)

[2.3.3. ArchProg.sql 15](#_Toc460097370)

[2.3.4. JobArchProg.sql 15](#_Toc460097371)

[2.4. MKT 15](#_Toc460097372)

[2.4.1. Modèle relationnel de données 15](#_Toc460097373)

[2.4.2. CreaMKT 16](#_Toc460097374)

[2.4.3. Package AlimMKT 16](#_Toc460097375)

[2.5. DW 18](#_Toc460097376)

[2.5.1. Modèle relationnel de données 18](#_Toc460097377)

[2.5.2. CreaDW 19](#_Toc460097378)

[2.5.3. Package AlimDW 19](#_Toc460097379)

[2.6. Description de l’application RechEvalFilm 19](#_Toc460097380)

* 1. Introduction

Ce dossier aura pour but d’expliquer, et donc de vous permettre de comprendre, toute la logique mise en œuvre pour le bon fonctionnement de notre application *Rennequinepolis* (que nous abrégerons en *RQS*).

Vous trouverez donc dans ce document un guide complet d’explications des différents packages et de l’application ainsi que les différentes explications qui s’apparentent à ceux-ci. De plus, tous les packages sont fournis et commentés eux aussi, ainsi que l’application RechEvalFilm.

* 1. Contexte général

RQS est une société qui distribue des films et qui gère les complexes cinématographiques projetant ces films. L’objectif de ce travail est d’informatiser leur système actuel et de construire les différentes plateformes web de consultation, d’achats et de gestion qu’ils souhaitent, ainsi que de réaliser les transferts d’informations nécessaires.

* + 1. Les différents sites

RQS est organisée en divers services, localisés à Bruxelles, en plus des complexes disséminés au travers du pays. Elle s’adresse à différents fournisseurs pour obtenir ses médias et en générer des copies qu’elle peut ensuite distribuer dans les différents cinémas.

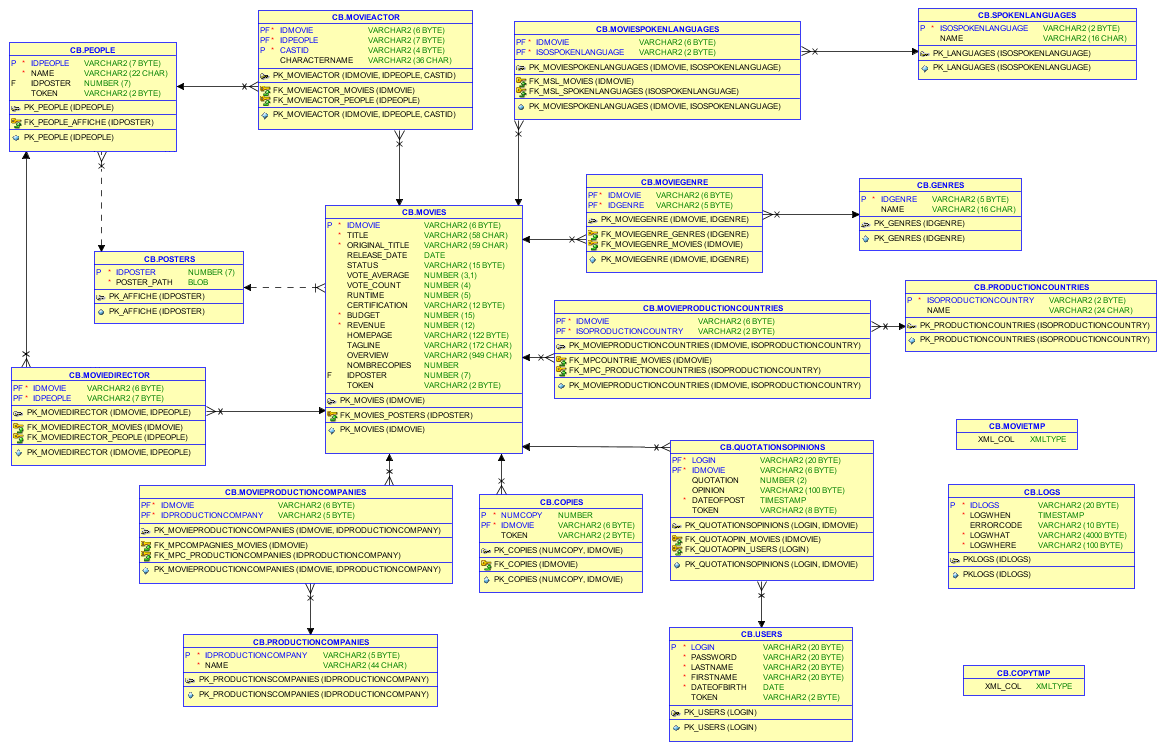
* **La centrale internationale – CI :** Cette centrale est un entrepôt où sont disponibles toutes les œuvres cinématographiques que pourraient souhaiter les entreprises comme RQS. Elle tient lieu de fournisseur principal de l’entreprise.
* **La centrale belge – CB :** Il s’agit de notre entreprise RQS. Située à Bruxelles, l’entreprise y dispose d’un entrepôt où elle peut stocker les copies avant qu’elles ne soient distribuées dans les cinémas. On y trouve également les serveurs principaux de RQS.
* **La centrale alternative belge – CBB :** Site de secours de RQS, il est capable de prendre le relais en cas de panne du site principal. Il propose les mêmes services que CB.
* **Les cinémas – CC :** Ce sont les différents complexes cinématographiques que gère RQS. Dans le cadre de cet énoncé, il n’y aura qu’un cinéma : CC.
* **Les personnes – BP\* :** C’est un répertoire des acteurs et réalisateurs des films de CI reprenant des informations plus complètes à leur sujet.
* **Le fournisseur de Ticket –** **TCKT\***: C’est un sous-traitant proposant des services de génération automatique de ticket aux complexes de cinémas.
* **Le centre d’analyse –** **DW/MKT** : C’est là que sont prises toutes les décisions financières les décisions les plus judicieuses en fonction des indicateurs reçus.

\*ces sites n’ont pas été mis en œuvre pour notre version de Rennequinepolis.

* + 1. Description des éléments de RQS
       1. SYS
          1. AccessControlList.sql

Création d’une ACL (Access Control List) permettant aux différentes bases de données de récupérer des informations provenant de sites internet. Cela sera utile pour le téléchargement des affiches de films stockés dans CB.

* + - * 1. CreateUsers.sql
* Création d’un rôle avec des privilèges.
* Création des utilisateurs avec attribution du rôle prédéfini au préalable.
* Création du directory avec autorisation, pour les utilisateurs en ayant besoin, d’aller le lire ou écrire.
  + - 1. CB/CBB
         1. Modèle relationnel de données



* + - * 1. CreaCB\_ScriptLDD.sql

Script de création des schémas CB et CBB. Ce script a été créé en fonction des résultats d’analyse statistique du package CreaCB.

* + - * 1. Package AlimCB

Ce package permet d’alimenter CB avec les films se trouvant dans la table Movies\_ext.

Procédure GetMoviesExt

IN : Nombre de films à ajouter.

OUT : /

Cette procédure permet de démarrer l’alimentation de CB. Attention, CI contenant près de 250.000 films, il ne faut pas tenter d’alimenter CB directement avec, par exemple, 5.000 films d’un coup, sous peine de devoir attendre très, très longtemps. De plus, en fonction des contraintes mises en place, il est possible que le nombre d’insertions ne soit pas de 5.000 mais bien moins (certains films ne respectent pas ces contraintes).

Procédure Parse

IN : Tableau de films à parser.

OUT : /

Cette procédure permet de vérifier si le film se trouve déjà dans CB ou non. Si celui-ci s’y trouve déjà, la procédure Parse appellera la procédure UpdateMovie pour ajouter un nombre de copies et passera au film suivant dans le tableau passé en paramètre, sinon elle appellera la procédure isMovieValid.

Fonction isMovieValid

IN : Le film à vérifier, un record du film à ajouter

OUT : Booléen

Cette fonction permet de tester si un film correspond bien aux critères attendus et respecte bien les contraintes mises en place. Pour se faire, elle va appeler plusieurs fonctions (ParseXXX) qui renvoient toutes un booléen afin de passer à la suivante. Si toutes ces fonctions ont renvoyé TRUE, isMovieValid renvoie elle aussi TRUE, et le film sera ajouté à CB.

Fonction ParseMovie

IN : Le film à vérifier, un record du film à ajouter

OUT : Booléen

Cette fonction permet de parser les données du film, de les nettoyer afin qu’elles respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseGenre, sinon on passe au film suivant.

Fonction ParseGenre

IN : Les genres du film à vérifier, un tableau des genres à ajouter

OUT : Booléen

Cette fonction permet de parser les genres du film, de les nettoyer afin qu’ils respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseActors, sinon on passe au film suivant.

Fonction ParseActor

IN : Les acteurs du film à vérifier, un tableau des acteurs à ajouter

OUT : Booléen

Cette fonction permet de parser les acteurs du film, de les nettoyer afin qu’ils respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseDirectors, sinon on passe au film suivant.

Fonction ParseDirector

IN : Les réalisateurs du film à vérifier, un tableau des réalisateurs à ajouter

OUT : Booléen

Cette fonction permet de parser les réalisateurs du film, de les nettoyer afin qu’ils respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseProductionCompanies, sinon on passe au film suivant.

Fonction ParseProductionCompanies

IN : Les compagnies de production du film à ajouter, un tableau des compagnies de production à ajouter

OUT : Booléen

Cette fonction permet de parser les sociétés de production du film, de les nettoyer afin qu’ils respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseProductionCountries, sinon on passe au film suivant.

Fonction ParseProductionCountries

IN : Les pays de production du film à vérifier, un tableau des pays de production à ajouter

OUT : Booléen

Cette fonction permet de parser les pays de production du film, de les nettoyer afin qu’ils respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, on passe à la fonction ParseSpokenLanguages, sinon on passe au film suivant.

Fonction ParseSpokenLanguages

IN : Les langues du film à vérifier, un tableau des langues à ajouter

OUT : Booléen

Cette fonction permet de parser les langues parlées du film, de les nettoyer afin qu’elles respectent bien les critères d’insertion. Elle renvoie un booléen. Si elle renvoie TRUE, la fonction isMovieValid renvoie TRUE et on commencera l’insertion des données, sinon on passe au film suivant.

Procédure AddMovie

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les données nettoyées du film, par la fonction ParseMovie, dans la table Movies.

Procédure UpdateMovie

IN : L’identifiant du film à mettre à jour, un nombre de copies

OUT : /

Cette procédure permet d’ajouter un nombre de copies à un film se trouvant déjà dans la base de données. Ce nombre de copies est basé sur une distribution normale de moyenne 5 et d’écart-type 2.

Procédure AddGenres

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les genres nettoyés du film, par la fonction ParseGenres, dans la table Genres, et de lié ces genres au film dans la table MovieGenre.

Procédure AddActors

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les acteurs nettoyés du film, par la fonction ParseActors, dans la table People, et de lié ces acteurs au film dans la table MovieActor.

Procédure AddDirectors

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les réalisateurs nettoyés du film, par la fonction ParseDirectors, dans la table People, et de lié ces acteurs au film dans la table MovieDirectors.

Procédure AddProductionCompanies

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les sociétés de production nettoyée du film, par la fonction ParseProductionCompanies, dans la table ProductionCompanies, et de lié ces sociétés au film dans la table MovieProductionCompanies.

Procédure AddProductionCountries

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les pays de production nettoyés du film, par la fonction ParseProductionCountries, dans la table ProductionCountries, et de lié ces pays au film dans la table MovieProductionCountries.

Procédure AddSpokenLanguages

IN : Le record du film à ajouter

OUT : /

Cette procédure permet d’insérer les langues parlées nettoyés du film, par la fonction ParseSpokenLanguages, dans la table SpokenLanguages, et de lié ces langues au film dans la table MovieSpokenLanguages.

* + - * 1. Package BackupCB

Ce package permet de faire une sauvegarde des informations du schéma CB sur le schéma CBB. Il permet donc d’avoir 2 schémas identiques et ainsi pouvoir basculer sur CBB et de pouvoir l’utiliser comme s’il s’agissait de CB.

Il sera ajouté aux 2 schémas, à savoir CB et CBB. Sur CB, il permettra la sauvegarde synchrone des cotes/avis des utilisateurs de RQS et la sauvegarde synchrone des copies de films se trouvant sur CB et CBB. De manière asynchrone, il sauvegardera tout ce qui ne se trouve pas encore sur CBB. Le package BackupCB se trouvant sur CBB sera alléger : il ne permettra que la sauvegarde synchrone des cotes/avis des utilisateurs de RQS vers CB. La sauvegarde dite « asynchrone » sera en fait une restauration qui sera faite par le package RestoreCB (voir le point 2.2.8 du document présent).

Fonction CheckUserToReplicate

IN : Le login de l’utilisateur à répliquer

OUT : Booléen

Cette fonction permet de vérifier si l’utilisateur existe sur CBB.

Fonction CheckMovieToReplicate

IN : L’identifiant du film à répliquer

OUT : Booléen

Ce fonction permet de vérifier si le film existe sur CBB.

Procédure Job\_ReplicationAsync

IN : /

OUT : /

Cette procédure permet de répliquer toutes les données de CB qui ne se trouvent pas sur CBB vers ce dernier. Pour cela, les différentes insertions se font en fonction des colonnes TOKEN des tables Users, Movies, Copies et QuotationsOpinions.

* + - * 1. Create\_Job.sql

Job lancé tous les jours à minuit qui effectuera la procédure Job\_ReplicationAsync du package BackupCB.

* + - * 1. Trigger\_ReplicationCopie.sql

Ce trigger est basé sur l’évènement BEFORE INSERT de la table Copies et est de type ligne (FOR EACH ROW). Il ne sera exécuté que si le tuple qui l’a déclenché contient un token à NULL. Il sert à répliquer de manière synchrone les copies de films vers CBB. Il appelle la fonction CheckMovieToReplicate afin de savoir si ce film se trouve déjà sur CBB ou non. S’il s’y trouve déjà, la copie sera répliquée vers CBB, sinon le token sera mis à « KO » et ce tuple sera répliqué de manière asynchrone via la procédure Job\_ReplicationAsync.

* + - * 1. Trigger\_ReplicationCotesAvis.sql

Ce trigger est basé sur les évènements BEFORE INSERT, DELETE et UPDATE (sur les colonnes Quotation et Opinion) de la table QuotationsOpinions et est de type ligne (FOR EACH ROW). Il ne sera exécuté que si le tuple qui l'a déclenché contient un token à NULL. Il sert à répliquer de manière synchrone les cotes/avis des utilisateurs de RQS vers CBB. Il appelle la fonction CheckUserToReplicate afin de savoir si cet utilisateur se trouve déjà sur CBB ou non. S’il s’y trouve déjà, les cotes/avis seront répliquées vers CBB, sinon le token sera mis à « KO » et ce tuple sera répliqué de manière asynchrone via la procédure Job\_ReplicationAsync.

* + - * 1. Package RestoreCB

Lorsque CB sera indisponible, CBB prendra la main afin de permettre à l’utilisateur de pouvoir continuer ses manipulations sur l’application « RechEvalFilm ». RestoreCB vérifie si CB est de nouveau. S’il l’est, ResotreCB va restaurer vers CB les données ajoutées sur CBB pendant que CB était HS.

Procédure CheckCB

IN : /

OUT : /

Cette procédure va vérifier si CB est de nouveau disponible ou s’il est toujours HS. Dans le cas où CB serait de nouveau disponible, cette procédure va appeler la procédure Restore (voir le point suivant) et lancer une exception utilisateur afin de prévenir que CB est de nouveau disponible. Cette exception sera captée et relancée par la procédure ou fonction appelante afin que l’application « RechEvalFilm » soit notifiée de la disponibilité de CB pour pouvoir se reconnecter à lui.

Procédure Restore

IN : /

OUT : /

Cette procédure va restaurer vers CB les nouvelles données ajoutées à CBB pendant que CB était indisponible. Les données restaurées ne sont que les nouveaux utilisateurs et toutes les cotes/avis ajoutées par ces nouveaux utilisateurs.

* + - * 1. Package CrashCB

Ce package permet de simuler la perte de connectivité à CB.

Il sera ajouté à l’utilisateur SYS. Toutes les commandes, hormis le SELECT de la procédure GetInfos, sont exécutées en SQL dynamique.

Procédure GetInfos

IN : Le login de l’utilisateur

OUT : /

Cette procédure permet de récupérer les SID et SerialNumber de l’utilisateur passé en paramètre (ici CB).

Procédure DisconnectSession

IN : Le login de l’utilisateur, un tableau de SID et SERIALNUMBER

OUT : /

Cette procédure permet de supprimer la (ou les) sessions de l’utilisateur passé en paramètre de la procédure GetInfos.

Procédure LockAccount

IN : Le login de l’utilisateur

OUT : /

Cette procédure permet de verrouiller un utilisateur (ici CB).

Procédure UnlockAccount

IN : Le login de l’utilisateur

OUT : /

Cette procédure permet de déverrouiller un utilisateur (ici CB).

* + - * 1. Package RechFilm

Ce package contient des fonctions permettant de rechercher diverses données selon des critères prédéfinis.

Fonction Connexion

IN : Le login de l’utilisateur

OUT : SYS\_REFCURSOR

Cette fonction permet de se connecter au schéma CB afin de pouvoir utiliser l’application RechEvalFilm.

Fonction GetMovie

IN : L’identifiant du film

OUT : SYS\_REFCURSOR avec le film recherché

Cette fonction permet de rechercher un film via son identifiant.

Fonction GetMovies

IN : Le titre, un/des acteurs, un/des réalisateurs, année de sortie, année de sortie antérieure, année de sortie postérieure du film

OUT : SYS\_REFCURSOR avec les films correspondants aux critères fournis.

Cette fonction permet de rechercher une liste de films sur base de plusieurs critères.

Fonction GetActorsFromMovie

IN : L’identifiant du film

OUT : SYS\_REFCURSOR avec les acteurs ayant joués dans le film recherché

Cette fonction permet de récupérer les acteurs ayant joué dans le film dont l’identifiant est passé en paramètre.

Fonction GetDirectorsFromMovie

IN : L’identifiant du film

OUT : SYS\_REFCURSOR avec les réalisateurs du film recherché

Cette fonction permet de récupérer les réalisateurs du film dont l’identifiant est passé en paramètre.

Fonction GetPoster

IN : L’identifiant du film

OUT : SYS\_REFCURSOR avec l’affiche du film recherché

Cette fonction permet de récupérer l’affiche du film dont l’identifiant est passé en paramètre.

Fonction GetQuotationsOpinionsFromRQS

IN : L’identifiant du film, numéro de page

OUT : SYS\_REFCURSOR avec le film recherché

Cette fonction permet de récupérer les avis laissés par les utilisateurs de RQS pour le film dont l’identifiant est passé en paramètre.

* + - * 1. Package EvalFilm

Procédure AddQuotationOpinion

IN : L’identifiant du film, le login de l’utilisateur, la cote, l’avis

OUT : /

Cette procédure permet d’insérer ou de modifier l’avis ou la cote qu’un utilisateur met à un film.

* + - * 1. Package AlimCC

Package permettant d’envoyer des copies de films sur CC.

Procédure MovieCopy

IN : L’identifiant du film

OUT : /

Cette procédure calcule un nombre aléatoire de copie. Si ce nombre est strictement supérieur à 0, on enregistre dans une table XML les données du film envoyé et dans une autre table XML les copies envoyées de ce film.

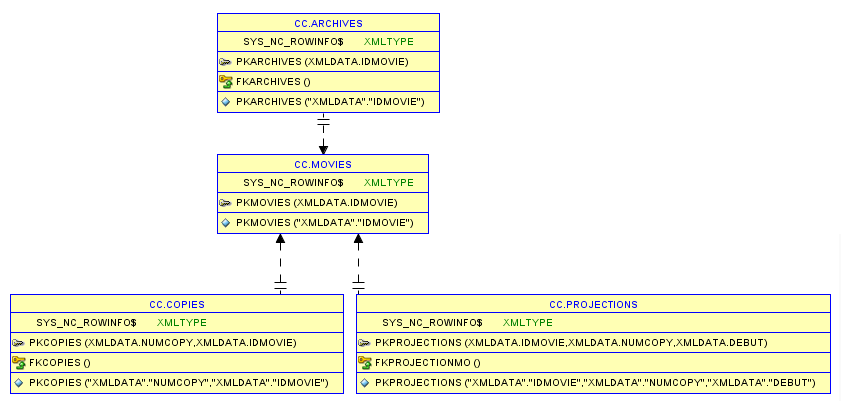
Procédure JobAlimCC

IN : /

OUT : /

Ce procédure permet d’envoyer des copies de chaque film sur CC toutes les semaines (procédure exécutée par un job). Elle appelle la méthode MovieCopy pour chaque film et appelle ensuite la procédure Reception sur CC.

* + - 1. CC
         1. Modèle relationnel de données



* + - * 1. CreaCC.sql

Fichier permettant d’enregistrer les différents schémas XML (\*.xsd) et de créer les tables XML composant le schéma CC.

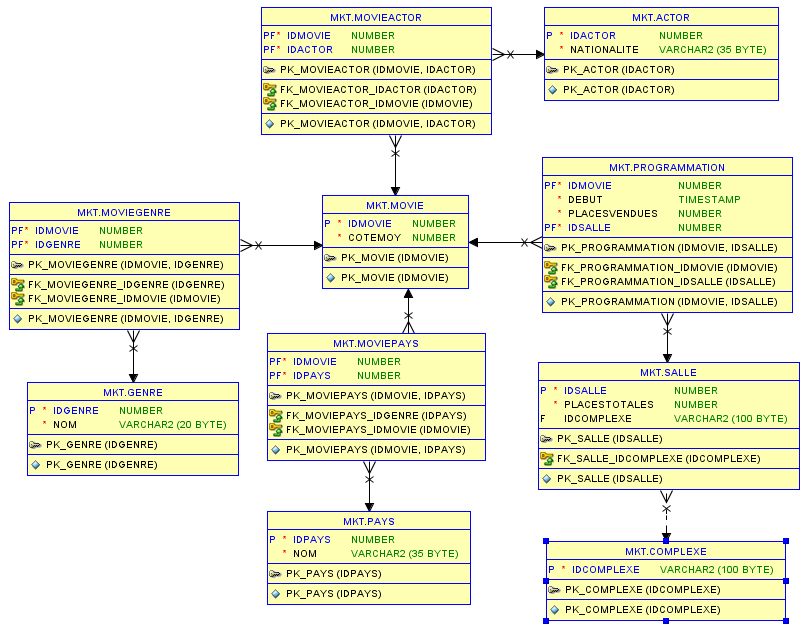
* + - * 1. ArchProg.sql

Procédure permettant d’archiver les programmations de films terminées.

* + - * 1. JobArchProg.sql

Job effectué tous les jours à minuit exécutant la procédure ArchProg.

* + - 1. MKT
         1. Modèle relationnel de données



* + - * 1. CreaMKT

Script LDD permettant de créer le schéma MKT.

* + - * 1. Package AlimMKT

Package regroupant toutes les procédures permettant d’alimenter le schéma MKT. Par manque de temps, et à cause de plusieurs problèmes provenant de fonctionnalités précédentes, toutes les données insérées dans le schéma MKT sont des données permettant de « bidonner » le schéma.

Procédure AlimentationMKT

IN : /

OUT : /

Cette procédure permet de lancer les différentes procédures composants le package AlimMKT.

Procédure MergeMovie

IN : /

OUT : /

Cette procédure permet de faire un MERGE sur la table Movie afin d’ajouter un film et sa cote, et de modifier sa cote si le film s’y trouve déjà.

Procédure InsertPays

IN : /

OUT : /

Cette procédure insère 30 pays dans le schéma MKT.

Procédure InsertActors

IN : /

OUT : /

Cette procédure insère 30 acteurs dans le schéma MKT.

Procédure InsertGenres

IN : /

OUT : /

Cette procédure insère 30 genres dans le schéma MKT.

Procédure MergeMoviePaysActorsGenres

IN : /

OUT : /

Cette procédure permet d’ajouter ou de mettre à jour les pays, acteurs et genres liés à une film.

Procédure MergeComplexe

IN : /

OUT : /

Cette procédure insère 30 complexes dans le schéma MKT.

Procédure InsertSalles

IN : /

OUT : /

Cette procédure permet de faire un MERGE sur la table Salle afin d’ajouter une salle avec un nombres de places totales dans un complexe, ou de modifier le nombre de places totales et le complexe si la salle s’y trouve déjà.

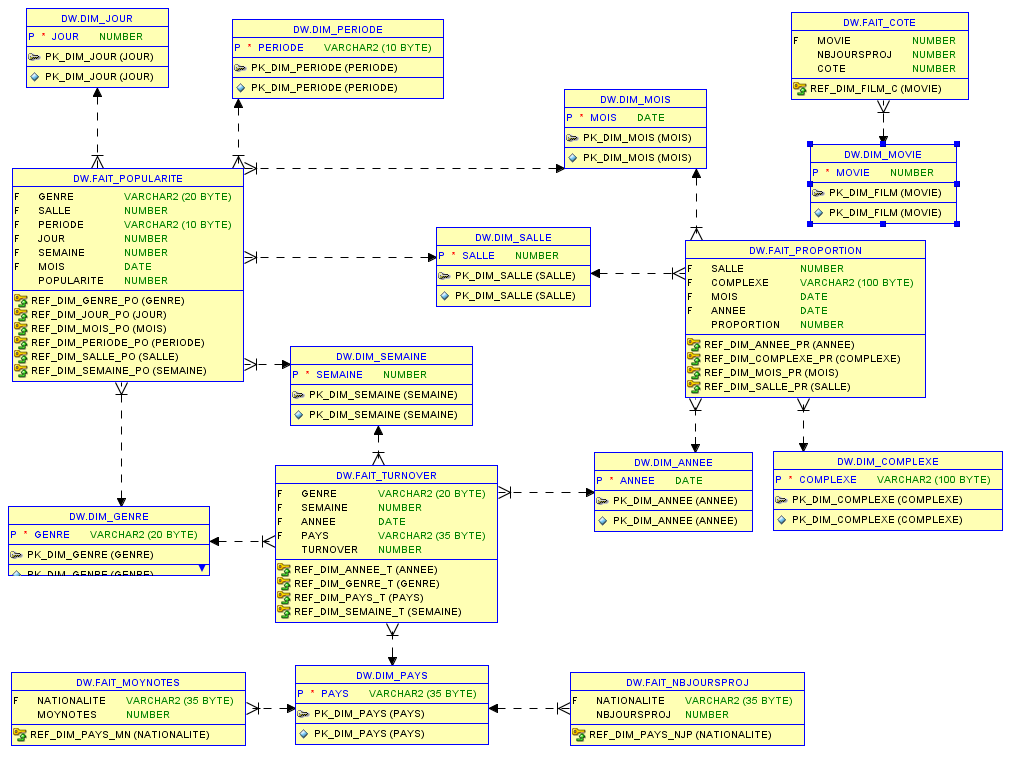
Procédure InsertProgrammations

IN : /

OUT : /

Cette procédure insère 30 programmation dans le schéma MKT.

* + - 1. DW
         1. Modèle relationnel de données



* + - * 1. CreaDW

Script LDD permettant de créer le schéma de diffusion de DW.

* + - * 1. Package AlimDW

Ce package regroupe les différentes procédures permettant d’alimenter les faits et les dimensions de DW.

Procédure Load

IN : /

OUT : /

Procédure permettant d’appeler les procédures LoadDimensions et LoadFaits.

Procédure LoadDimensions

IN : /

OUT : /

Procédure permettant d’alimenter les dimensions à partir des données se trouvant sur MKT.

Procédure LoadFaits

IN : /

OUT : /

Procédure permettant d’alimenter les faits à partir des données se trouvant sur MKT.

* + - 1. Description de l’application RechEvalFilm

L’application proprement dite utilise 2 packages (Java) pour fonctionner :

* Le package RechFilm qui contient toutes les fonctionnalités de recherche de films que ce soit sur son id ou bien sur d’autres critères tels que son titre, son année de sortie, …
* Le package EvalFilm qui permet de donner une cotation et un avis sur un film recherché.

L’application a été écrite en Java et utilise un bean pour faire les accès à la base de données.

Celle-ci contient deux packages rechEvalFilm et Others. Le premier est celui qui premet de faire tourner l’application tandis que l’autre regroupe deux fonctionnalités : ImageRenderer qui permet l’affichage des images dans l’application et ThreadCheckDatabase qui permet de tester si la base de données et toujours accessible.