MAHIER Loïc
JEHANNO Clément
JAMET Félix
PHALAVANDISHVILI Demetre

groupe 601B

Rapport préliminaire de projet *

*rapport réalisé sous LaTeX

Sommaire

1. Introduction

Dans le cadre de ce projet nous devons créer une base de données. Nous avons décidé de modéliser la gestion de cinémas sur une grande échelle. Par exemple nous voulons savoir quels sont les cinémas de France, à qui ils appartiennent (Pathé,UGC, etc.) et ce qu'ils proposent. Comme notre modèle se base sur une certaine réalité, voici comment nous avons décomposé la chose, prenons l'exemple d'un cinéma :

Le cinéma Pathé à Atlantis, dans la ville de Nantes. Tout d'abord on voit que un cinéma est identifié par une adresse et une ville. Ensuite, notre cinéma possède des salles dans lesquelles seront diffusés des films. Chaque film est composé d'une équipe d'acteurs, d'un réalisateur et d'une date de sortie. Il peut être compatible ou non avec la 3D.

En ce qui concerne nos salles, elles possèdent un certain nombre de places qui sont réparties entre les places "standard", les places "handicapés" ainsi que les nouveaux sièges dBox (sièges bougeant en même temps que le film). Si elles sont compatibles, elles ont la possibilité de diffuser en 3D.

Une séance dans un cinéma donné correspond à une date de projection d'un certain film dans une salle spécifique à un horaire précis, le film étant diffusé ou non en 3D. Aujourd'hui si on va au cinéma il est possible de réserver sa séance, autrement dit, on réserve un certain nombre de places pour un film, à un horaire précis, dans un cinéma donné.

2. Première partie

2.1 Répartition des tâches

Voici comment nous nous sommes organisés pour répartir les tâches :

Tout d'abord après les premières semaines de cours nous nous sommes réunis pour décider ensemble d'un sujet. L'idée du cinéma est venue assez naturellement et nous paraissait être à la fois concrète et proche de la réalité.

Ensuite nous avons défini tous les attributs de notre table, en se demandant ensemble : Que voulons-nous faire ? Comment voulons nous le faire ? Un cinéma fonctionne t-il vraiment comme ça ? L'ajout de tel ou tel attribut est-il pertinent ? etc etc. Une fois nos attributs répartis nous avons chacun

pris un cinéma (on en a 4) et chaque personne a remplit la partie du tableau qui correspondait à un cinéma. Après cela on a observé nos tuples sur tableur et on a relevé nos dépendances fonctionnelles.

Ensuite Demetre et Félix ont fait l'algorithme de décomposition et Loïc et Clément ont fait l'algorithme de Bernstein. On a mis en commun le résultat des deux algorithmes afin de voir si on avait la même chose ou non, et pourquoi. Pour finir, nous avons testé la normalisation de notre schéma avec l'outil mis à notre disposition en question 5.

2.2 Table de base

Vous trouverez en annexe la table (5) contenant tous nos attributs ainsi que tous nos tuples. Celle-ci est en quatre parties à cause de sa taille conséquente.

2.3 Dépendances fonctionnelles

A partir de la table ci-dessous contenant tous nos attributs, nous avons déduit les douze dépendances fonctionnelles ci-dessous. Pour ce faire nous avons ajouté des attributs (id) nous permettant de simplifier nos relations.

- (1) idCine → adresse, ville
- (2) adresse, ville → franchise, nbSalle
- (3) idCine → franchise, nbSalles
- (4) idCine, numSalle → salleCompatibleEn3D, nbPlaceStandard, nbPlaceHandicape,nbDbox
- (5) idFilm → nomFilm, dateSortie
- (6) nomFilm, dateSortie → public, idReal, duree, compatible3D
- (7) idFilm, role → idAct
- (8) idReal → nomR, prenomR
- (9) idAct → nomA, prenomA
- (10) idClient → nomC, prenomC
- (11) idClient, numReservation → nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance
- (12) idSeance → idCine, horaire, dateProjection, numSalle, idFilm, diffusionEn3D

Avec les dépendances fonctionnelles ci-dessus, nous obtenons le graphe des dépendances en annexe (9). De cela nous déterminons la clé suivante : {idCine, idClient, numReservation, role}.

2.4 Algorithme de Bernstein

L'algo de Bernstein se fait en 4 parties :

- ➤ Calculer la CV(DF) et les clés. Si R est en 3FN on s'arrête.
- ➤ Partitionner CV(DF) en groupe DFi (1 <= i <= k) tel que toutes les dfs d'un même groupe aient la même partie gauche.
- ➤ Construire un schéma <Ri(Ui), DFi> pour chaque groupe DFi, où Ui est l'ensemble des attributs apparaissant dans DFi.
- ➤ Si aucun des schémas définis ne contient de clé X de R, rajouter un schéma <Rk+1(X), {}>.

2.4.1 Calcul de CV(DF)

La couverture minimale se fait en trois parties :

- ➤ Toutes les dépendances doivent être élémentaires ; les décomposer si nécessaire.
- ➤ Eliminer les attributs superflus du coté gauche de la df.
- ➤ Eliminer les dfs redondantes.

Pas 1

On décompose chacune des dfs en dfe :

- (1) idCine → ville
- (1) idCine → adresse
- (2) adresse, ville → franchise
- (2) adresse, ville → nbSalle
- (3) idCine → franchise
- (3) idCine → nbSalles
- (4) idCine, numSalle → salleCompatibleEn3D
- (4) idCine, numSalle → nbPlaceStandard

- (4) idCine, numSalle → nbPlaceHandicape
- (4) idCine, numSalle → nbDbox
- (5) idFilm → nomFilm
- (5) idFilm → dateSortie
- (6) nomFilm, dateSortie → public
- (6) nomFilm, dateSortie → idReal
- (6) nomFilm, dateSortie → duree
- (6) nomFilm, dateSortie → compatible3D
- (7) idFilm, role → idAct
- (8) idReal → nomR
- (8) idReal → prenomR
- (9) idAct → nomA
- (9) idAct → prenomA
- (10) idClient → nomC
- (10) idClient → prenomC
- (11) idClient, numReservation → idSeance
- (11) idClient, numReservation → nbPlaceStandardRes
- (11) idClient, numReservation → nbPlaceHandicapeRes
- (11) idClient, numReservation → nbPlaceDBoxRes
- (12) idSeance, idCine → horaire
- (12) idSeance, idCine → dateProjection
- (12) idSeance, idCine → numSalle
- (12) idSeance, idCine → idFilm
- (12) idSeance idCine → diffusionEn3D

Pas 2

On prend toutes les dfs qui ont plus d'un attribut à gauche et on calcul leur fermeture. On élimine l'autre attribut si l'attribut de droite de la df apparaît dans le résultat, ou s'il apparaît dans le résultat de la fermeture.

- (2) adresse, ville → franchise, nbSalle

adresse+
adresse
ville+
ville
→ Les deux attributs sont nécessaires.

- (4) idCine, numSalle → salleCompatibleEn3D, nbPlaceStandard, nbPlaceHandicape,nbDbox

idCine+
idCine / adresse / ville /franchise / nbSalle
numSalle+
numSalle
→ Les deux attributs sont nécessaires.

- (6) nomFilm, dateSortie → public, idReal, duree, compatible3D

nomFilm+
nomFilm
dateSortie+
dateSortie
→ Les deux attributs sont nécessaires.

- (7) idFilm, role → idAct

idFilm+
idFilm / nomFilm / dateSortie / public / idReal / duree / compatible3D / nomA / prenomA
role+
role
→ Les deux attributs sont nécessaires.

- (11) idClient, numReservation → nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance

idClient+
idClient / nomC / prenomC
numReservation+
numReservation
→ Les deux attributs sont nécessaires.

- (12) idSeance, idCine → horaire, dateProjection, numSalle, idFilm, diffusionEn3D

idSeance+
idSeance

idCine+ adresse / ville / franchise / nbSalle
→ Les deux attributs sont nécessaires.

Pas 3

Eliminons tout d'abord les dfs qui sont préservées par transitivité :

- (1) idCine → adresse, ville
- (2) adresse, ville → franchise, nbSalle
- (3) idCine → franchise, nbSalles

Si l'on prend les dfs 1, 2 et 3, on remarque que l'on peut supprimer la 3 car on peut retrouver celle-ci par transitivité. Reprenons donc nos dfs restantes :

- (1) idCine → adresse, ville
- (2) adresse, ville → franchise, nbSalle
- (3) idCine, numSalle → salleCompatibleEn3D, nbPlaceStandard, nbPlaceHandicape,nbDbox
- (4) idFilm → nomFilm, dateSortie
- (5) nomFilm, dateSortie → public, idReal, duree, compatible3D
- (6) idFilm, role → idAct
- (7) idReal → nomR, prenomR
- (8) idAct → nomA, prenomA
- (9) idClient → nomC, prenomC
- (10) idClient, numReservation → nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance
- (11) idSeance, idCine → horaire, dateProjection, numSalle, idFilm, diffusionEn3D

A présent, analysons chaque dépendance fonctionelle une par une :

- (1) idCine → adresse, ville

idCine+
idCine
→ La df est préservée.

- (2) adresse, ville → franchise, nbSalle

adresse, ville+
adresse, ville
→ La df est préservée.

- (3) idCine, numSalle → salleCompatibleEn3D, nbPlaceStandard, nbPlace-Handicape,nbDbox

idCine, numSalle+
idCine / adresse / ville / franchise / nbSalle / numSalle
→ La df est préservée.

- (4) idFilm → nomFilm, dateSortie

idFilm+
idFilm
→ La df est préservée.

- (5) nomFilm, dateSortie → public, idReal, duree, compatible3D

nomFilm, dateSortie+
nomFilm / dateSortie
→ La df est préservée.

- (6) idFilm, role → idAct

idFilm, role+
idFilm / nomFilm / dateSortie / public / idReal / duree / compatible3D / nomR / prenomR / role
→ La df est préservée.

- (7) idReal → nomP, prenomP

idReal+
idReal
→ La df est préservée.

- (8) idAct → nomP, prenomP

idAct+

idAct
→ La df est préservée.

- (9) idClient → nomC, prenomC

idClient+
idClient
→ La df est préservée.

- (10) idClient, numReservation → nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance

idClient, numReservation+
idClient / nomC / prenomC / numReservation
→ La df est préservée.

- (11) idSeance, idCine → horaire, dateProjection, numSalle, idFilm, diffusionEn3D

idSeance, idCine+
idSeance / idCine / adresse / ville / franchise / nbSalle
→ La df est préservée.

Ainsi, hormis la suppression d'une dépendance fonctionnelle transitive, nos dépendances fonctionelles ne changent pas.

On constate que l'on est bien en 1FN, ainsi qu'en 2FN. Cependant nous ne sommes pas en 3FN. En effet, nous avons des attributs non clés, qui déterminent d'autres attributs non clés. Par exemple, adresse et ville sont deux attributs non clés qui déterminent franchise et nbSalle qui sont eux aussi non clés (df(2)).

2.4.2 Partitionnement de la CV et construction des schémas

R1 = {idCine, adresse, ville}
DF1 = {idCine → adresse, ville}

R2 = {adresse, ville, franchise, nbSalle}
DF2 = {adresse, ville → franchise, nbSalle}

R3 = {idCine, numSalle, salleCompatibleEn3D, nbPlaceStandard, nbPlaceHandicape, nbDbox}
DF3 = {idCine, numSalle, → salleCompatibleEn3D, nbPlaceStandard, nbPlaceHandicape, nbDbox}

R4 = {idFilm, nomFilm, dateSortie}
DF4 = {idFilm → nomFilm, dateSortie}

R5 = {nomFilm, dateSortie, public, idReal, duree, compatible3D}
DF5 = {nomFilm, dateSortie → public, idReal, duree, compatible3D}

R6 = {idFilm, role, idAct}
DF6 = {idFilm, role → idAct}

R7 = {idReal, nomR, prenomR}
DF7 = {idReal → nomR, prenomR}

R8 = {idAct, nomA, prenomA}
DF8 = {idAct → nomA, prenomA}

R9 = {idClient, nomC, prenomC}
DF9 = {idClient → nomC, prenomC}

R10 = {idClient, numReservation, nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance}
DF10 = {idClient, numReservation → nbPlaceStandardRes, nbPlaceHandicapeRes, nbPlaceDBoxRes, idSeance}

R11 = {idSeance, idCine, horaire, dateProjection, numSalle, idFilm, diffusionEn3D}
DF11 = {idSeance, idCine → horaire, dateProjection, numSalle, idFilm, diffusionEn3D}

On constate que nous n'avons pas de relation contenant toutes nos clés, c'est pourquoi nous devons créer une relation pour cela, avec une dépendance fonctionnelle associée vide.

R12 = {idCine, idClient, numReservation, role}
DF12 = {}

2.5 Algorithme de décomposition

Nous avons pris tous nos attributs puis nous avons suivi l'algorithme de décomposition. C'est à dire que en fonction de nos attributs et de nos dépendances fonctionnelles, nous avons pris une dépendance et avons fait une relation en fonction de cette dépendance. Puis nous avons retiré nos attributs non clés du reste de la relation originale. Ensuite on a recommencé jusqu'à arriver à une partie ne contenant que des clés et plus de dépendance fonctionnelle.

On a commencé par les dépendances fonctionelles qui n'avaient qu'un attribut à gauche puis nous avons fini par celles qui avaient plusieurs attributs à gauche.

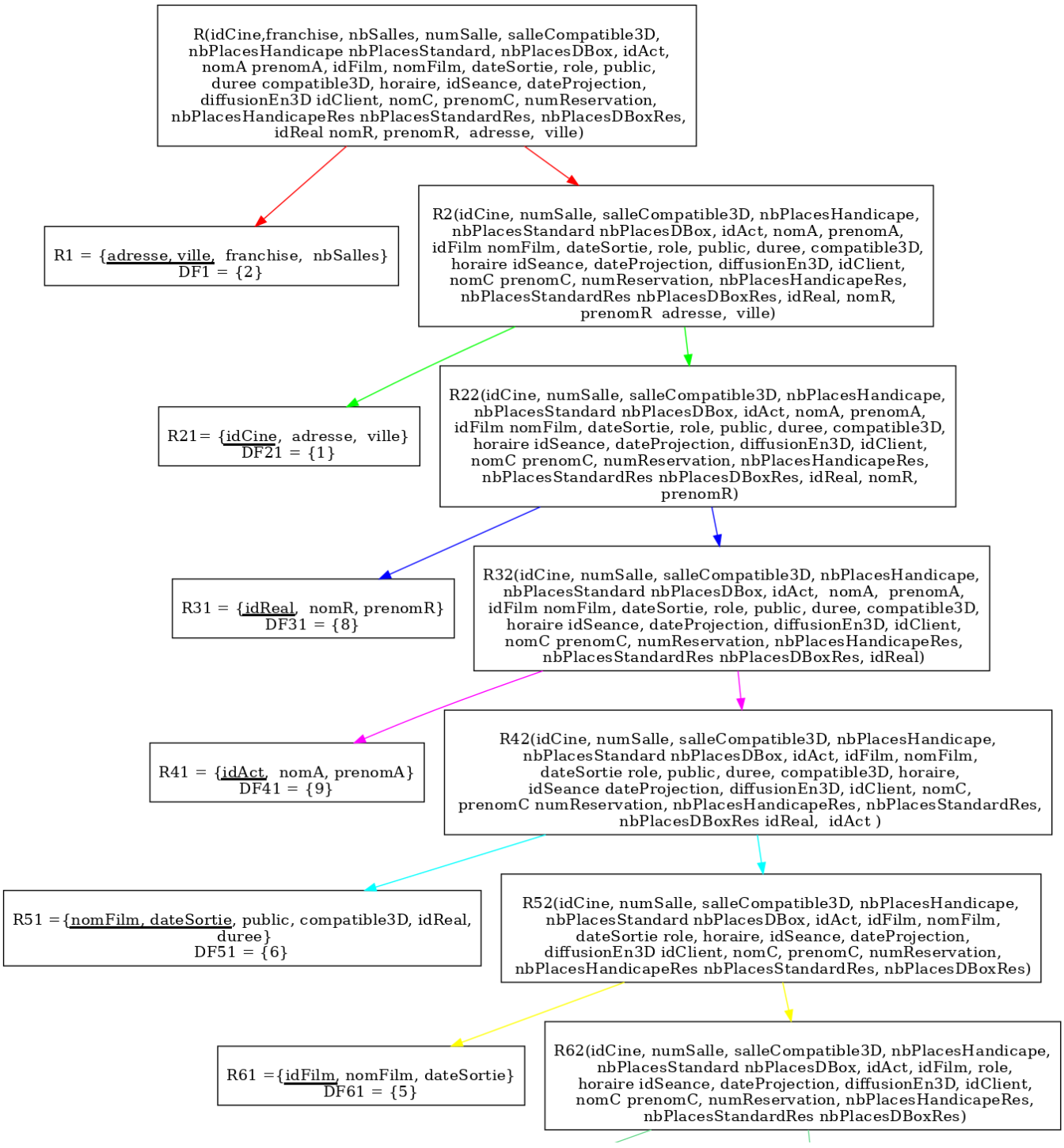

FIGURE 1 – Première partie de notre algorithme de décomposition

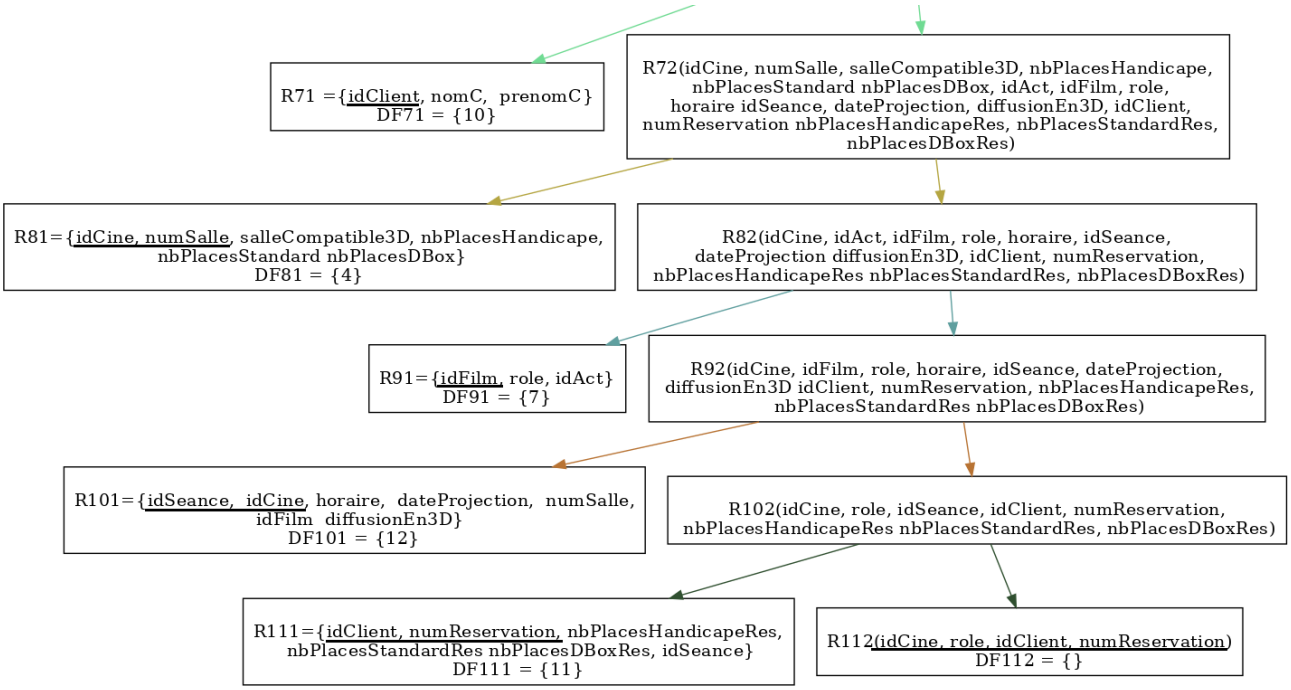

FIGURE 2 – Deuxième partie de notre algorithme de décomposition

2.6 Schéma de nos tables

Pour le diagramme nous nous sommes basés sur le résultat de l'algorithme de décomposition qui nous a fourni des tables équivalentes aux 11 relations R1..R11. Cependant nous avons fusionné deux fois deux tables : la table R1 et R21 car cela nous paraît plus logique d'avoir pour chaque adresse, ville, franchise et nbSalles un seul idCine qui détermine un unique cinéma.

Cette table devient donc la table Cinema, ce qui donne plus de sens à notre modèle. Nous avons aussi fusionné R51 et R61 pour en faire une seule et même table Film pour les mêmes raisons.

Pour finir on a renommé toutes les autres tables pour leur donner un nom plus parlant : R31 est devenue Realisateur, R41 est devenue Acteur, R71 est devenue Client, R81 est devenue Salle, R91 est devenue Casting, R101 est devenue Seance et 111 est devenue Reservation.

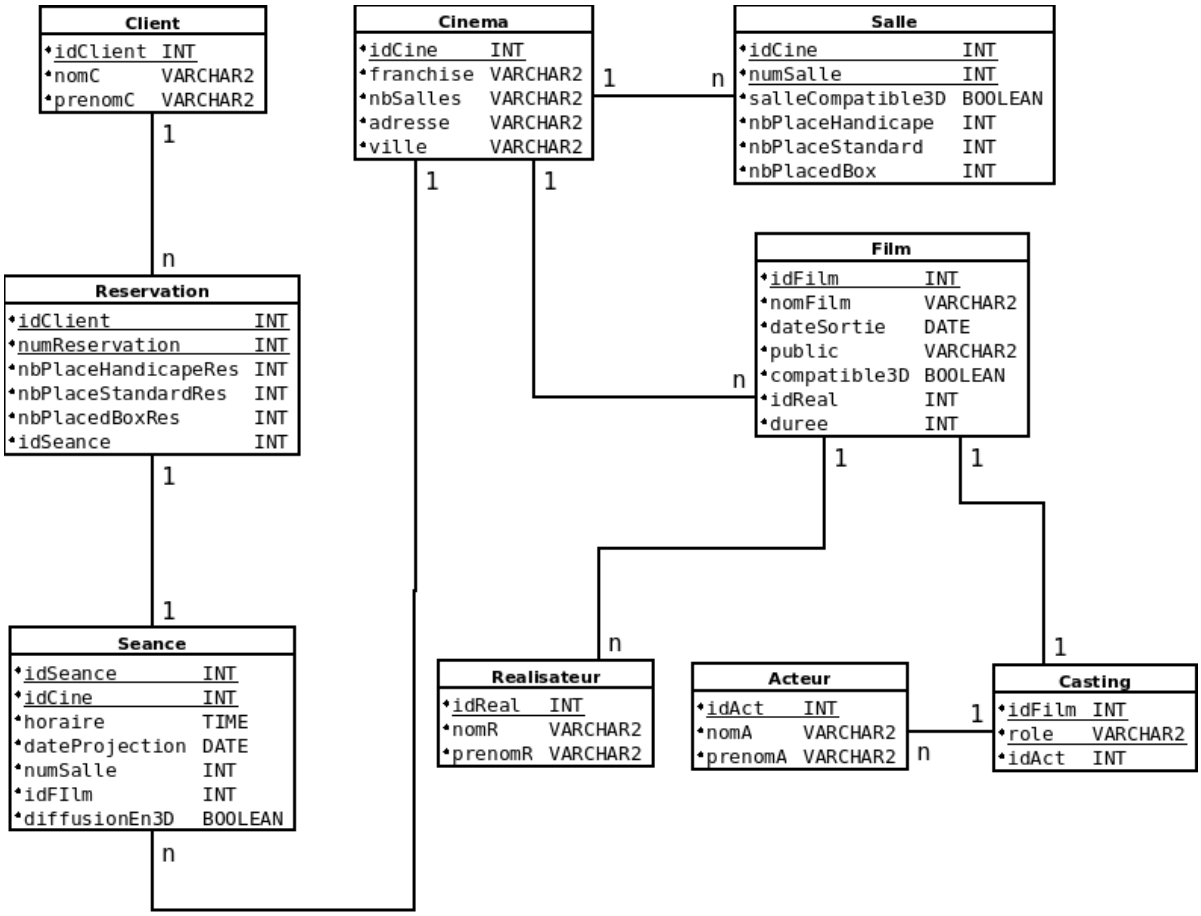

FIGURE 3 – Diagramme UML de nos tables

2.7 Conclusion préliminaire

Cette première partie nous a permis de bien poser les bases de notre base de donnée. De plus, l'algorithme de décomposition nous permet d'obtenir des tables qui nous semblent cohérentes avec la réalité. Pour la suite, nous avons déjà quelques idées de contraintes, de triggers et de fonctions qu'il nous faudra intégrer pour bien gérer nos cinémas.

3. Deuxième partie

3.1 Modifications de nos schémas

Suite au dépôt de la première partie du projet concernant principalement nos tables, nous avons remarqué quelques incohérences. Nous allons commencer cette deuxième partie du rapport par corriger ces défauts, ainsi nous pourrons continuer sur une base saine. Tout d'abord, avec le schéma précédemment obtenu, un film ne pouvait avoir plusieurs rôles, ce qui est incorrect. Nous avons donc rectifiés la cardinalité entre les tables Casting et Film. D'autre part, nous avions un problème similaire entre les tables Séance et Réservation. En effet on ne pouvait avoir qu'une réservation par séance. Ensuite, et c'est peut-être le problème majeur, nous avions une incohérence au niveau de nos clés primaires pour la table Séance, ce qui bloquait la jointure naturel entre Séance et Réservation. Comme idCine était clé primaire dans Séance, il aurait fallu ajouter dans Réservation la clé idCine pour pouvoir faire une jointure naturel. Cependant, nous avons préféré enlever idCine de la clé primaire de la table Réservation, pour la raison que nous ne voulons pas qu'une réservation soit propre au cinéma, mais globale.

Enfin, en réfléchissant sur le type de variable de l'attribut horaire, on s'est rendu compte que le type "TIME" était obsolète. Ainsi, au lieu d'avoir deux variables dateProjection et horaire, nous avons décidé d'avoir une seule variable qui contient à la fois la date et l'heure de la projection. Aussi nous avons toujours la possibilité de faire des comparaisons car ce format de date nous le permet.

Voici le nouveau diagramme UML :

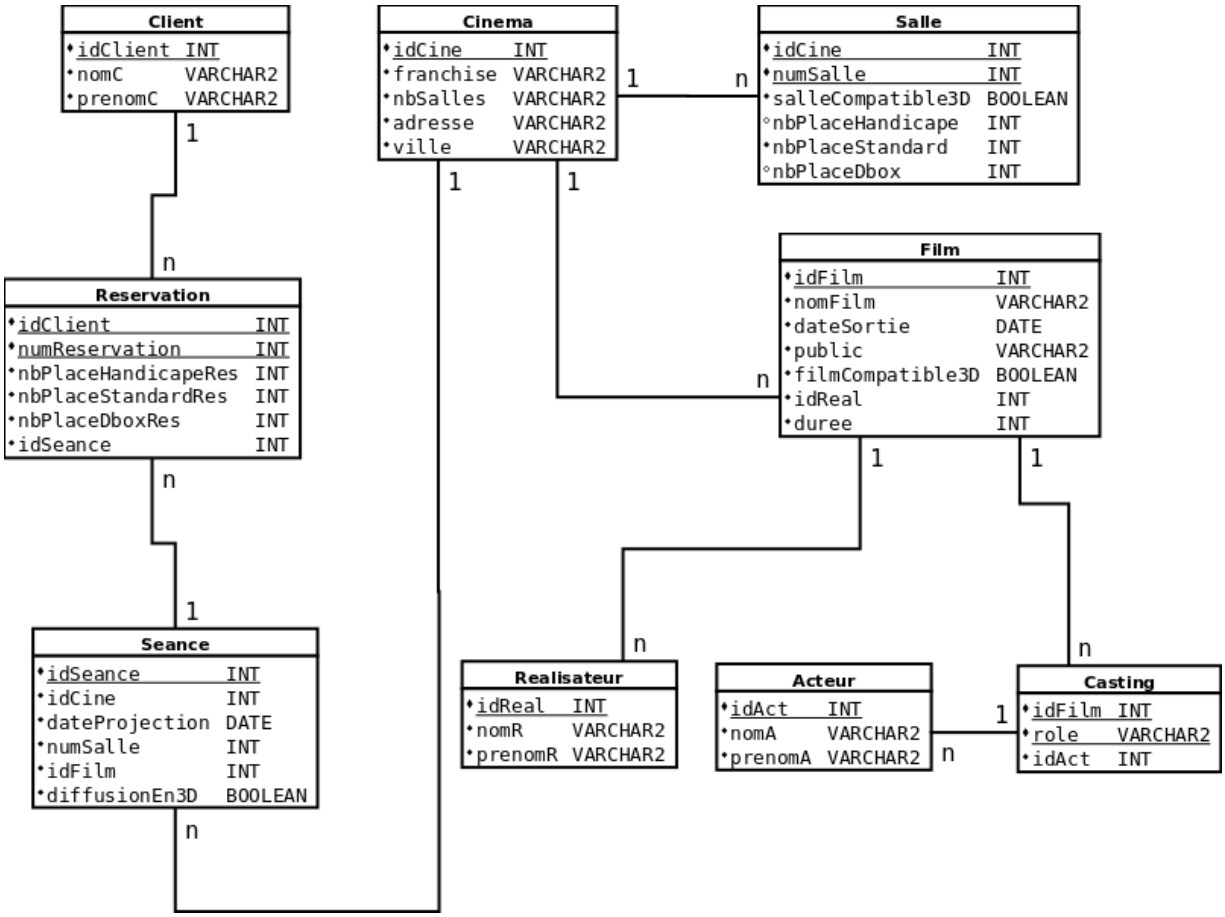

FIGURE 4 – Nouveau diagramme UML de nos tables

3.2 Répartition des tâches

Au début, nous nous sommes réunis tous les quatre pour discuter des problèmes qui se posaient suite à la première partie et apporter les modifications nécessaires (voir ci-dessus). Ensuite, on s'est répartis les tâches comme suit : Demetre a travaillé sur les triggers, Loïc a fait les vues et une procédure à l'aide de Clément et enfin, Félix a fait les insertions de tuples ainsi qu'une procédure. Pour finir, Loïc et Clément ont rédigés le rapport pendant que Demetre et Félix faisaient les jeux d'essais.

3.3 Explications

3.3.1 Les triggers

➤ verif3D :

Le trigger verif3D a pour rôle de s'assurer qu'une séance diffusée en 3D est projetée dans une salle compatible avec la projection en 3D. Une exception est levée si la séance est en 3D mais que la salle n'est pas compatible 3D. Ainsi, une salle compatible 3D peut parfaitement projeter des films non 3D.

➤ verifPlace :

Avant d'insérer ou de faire une modification sur Reservation on vérifie qu'on excède pas la capacité de la salle. On va donc calculer toutes les places qui sont réservées, on additionne le nombre de places déjà réservées avec le nombre de places qu'on veut réserver et si on excède la capacité de la salle alors on lève une exception suivant le nombre de place qui pose problème. En effet, on distingue chaque type de place, une personne peut réserver 10 places standard et 1 place dbox, si il y a 0 place dbox mais 50 places standard il faut lui dire qu'il n'y a pas assez de places dbox mais assez de places standard.

3.3.2 Les vues

➤ seanceDisponible :

Cette vue a été pensée pour le client, en effet elle indique les séances disponibles à partir de la date du système pour lesquelles il reste encore des places à réserver. Ainsi elle retourne le titre du film, la date de projection, le nom du cinéma et la salle.

➤ seanceAVenir :

Cette vue nous permet de faire un zoom sur les séances qui n'ont pas encore eu lieu. On récupère donc tout sur la table Seance avec pour

condition d'avoir une date ultérieure à la date actuelle. Cette vue nous a été utile dans le cadre d'une fonction de réservation.

3.3.3 Les procédures

➤ reservationClient :

La procédure reservationClient prend en paramètre un nom, un prénom et une séance ainsi que le nombre de places qu'on veut réserver. Comme quand on va au cinéma, si on veut réserver on précise combien de places on prend. Le nom et le prénom nous servent à savoir si le Client est déjà dans la base ou non. On a donc deux tests : déjà on veut savoir si on peut réserver c'est ici qu'on se sert de notre vue : on regarde si la séance est bien à venir et qu'elle n'est pas déjà passée. Une fois que c'est fait on regarde si le client est dans la base : si il l'est, on récupère son id et on ajoute directement sa réservation, sinon on l'ajoute dans la base des clients et on fait ensuite sa réservation avec le nouvel identifiant que l'on vient de lui attribuer.

➤ affichPlaces :

Cette fonction prend en paramètre l'identifiant d'une séance et affiche le nombre de places standard, handicapé et dbox restantes. Elle pourrait être utile à un guichetier voulant s'assurer que les places qu'il s'apprête à vendre sont disponibles. Dans un premier temps, cette fonction calcule pour chaque type de place : le nombre ayant été réservé pour la séance fournie en paramètre et le nombre de places dans la salle dans laquelle la séance a lieu (la capacité de la salle). Le nombre de places restante est la capacité de la salle à laquelle on soustrait le nombre de places réservées.

3.3.4 Les droits

Avant toute chose, on se place dans un contexte d'utilisation externe de la base de données, par exemple dans le cas où nous arrivons à un cinéma et nous pouvons donc consulter une programmation ou traiter avec des guiche-

tiers.

➤ lesClients :

On simule ici les droits que possède un client, ou plus simplement un utilisateur du site web. Le client aura seulement les droits de lecture sur la table ainsi il pourra rechercher le titre d'un film ou bien même un cinéma. Ils ont donc un accès en lecture sur les tables Seance, Cinema et Film.

➤ lesGuichetiers :

Nos guichetiers sont ceux qui vont réserver les places ils ont donc un accès en lecture sur les tables Seance et Reservation mais aussi un droit de modification (en écriture) sur Reservation. Cependant, il doit vérifier si le client est dans la base donc on lui permet aussi un droit de sélection sur la table Client, si il n'y est pas il peut l'ajouter donc on lui donne un droit de modification.

➤ lesGerantsFilm :

Les gérants des films et donc de la programmation vont agencer les films et les séances pour le cinéma. Ils ont un droit de lecture sur les films et les salles mais aussi de modification sur les séances. Ainsi ils peuvent créer leurs séances pour gérer leur cinéma.

3.3.5 Les index

Les clés primaires étant déjà indexées par défaut, nous avons rajouté deux index d'attributs non primaires qui nous semblaient utiles. Cette fois ci, on se place dans le cas d'un client qui fait une recherche sur internet.

➤ nomFilm :

La plupart des clients savent quel film ils vont voir, ils cherchent donc à savoir où et quand est diffusé ce film. Ceci est un exemple pour montrer que l'attribut titre est un attribut qui sera beaucoup sollicité, son indexation permettra donc de gagner du temps sur nos requêtes et donc

que la recherche du client soit plus rapide.

- ville :

  De même, il nous semble pertinent de savoir quels cinémas se trouvent dans notre ville.

3.4 Atouts, améliorations envisageables et critique

4. Conclusion

5. Annexe

idCine	adresse	ville	franchise	nbSalle	numeroSalle	Salle compatible 3D	nbPlaceStandard
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
1	Place jean-bart	St-Herblain	UGC	10	7	oui	100
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
2	Allee la perouse	St-Herblain	Pathé Gaumont	12	11	oui	255
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
3	12 Place du commerce	Nantes	Pathé Gaumont	8	1	non	345
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75
4	1ere avenue des droits de l'homme	Angers	Pathé Gaumont	5	3	non	75

FIGURE 5 – Première partie de notre table contenant tous les attributs et quelques tuples

nbPlaceHandicape	nbDbox	idFilm	nomFilm	dateSortie	duree	public	filmCompatible3D	idRea	nomR
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
3	0	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
5	10	1	xXx : Reactivated	19/01/17	110	Tout public	oui	1	Caruso
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
10	10	2	Assassin's Creeds	21/12/16	140	-12	oui	2	Kurzel
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle
3	3	3	Lalaland	25/01/17	125	Tout public	non	3	Chazelle

FIGURE 6 – Deuxième partie de notre table contenant tous les attributs et quelques tuples

nomR	prenomR	idAct	nomA	prenomA	rôle	idSeance	dateProjection	horaire	diffusionEn3d	idClient
Caruso	Daniel John	1	diesel	vin	xander cage	1	24/01/17	20h	oui	1
Caruso	Daniel John	2	yan	donnie	xiang	1	24/01/17	20h	oui	1
Caruso	Daniel John	3	padukone	deepika	serena unger	1	24/01/17	20h	oui	1
Caruso	Daniel John	4	wu	kris	nicks	1	24/01/17	20h	oui	1
Caruso	Daniel John	5	rose	ruby	adele wofl	1	24/01/17	20h	oui	1
Caruso	Daniel John	6	McCann	rory	tennyson torch	1	24/01/17	20h	oui	1
Caruso	Daniel John	1	diesel	vin	xander cage	1	24/01/17	20h	oui	2
Caruso	Daniel John	2	yan	donnie	xiang	1	24/01/17	20h	oui	2
Caruso	Daniel John	3	padukone	deepika	serena unger	1	24/01/17	20h	oui	2
Caruso	Daniel John	4	wu	kris	nicks	1	24/01/17	20h	oui	2
Caruso	Daniel John	5	rose	ruby	adele wofl	1	24/01/17	20h	oui	2
Caruso	Daniel John	6	McCann	rory	tennyson torch	1	24/01/17	20h	oui	2
Caruso	Daniel John	1	diesel	vin	xander cage	2	27/01/17	16h30	Non	3
Caruso	Daniel John	2	yan	donnie	xiang	2	27/01/17	16h30	Non	3
Caruso	Daniel John	3	padukone	deepika	serena unger	2	27/01/17	16h30	Non	3
Caruso	Daniel John	4	wu	kris	nicks	2	27/01/17	16h30	Non	3
Caruso	Daniel John	5	rose	ruby	adele wofl	2	27/01/17	16h30	Non	3
Caruso	Daniel John	6	McCann	rory	tennyson torch	2	27/01/17	16h30	Non	3
Kurzel	Justin	7	fassbender	michael	cal lynch	3	10/01/17	11h00	Non	4
Kurzel	Justin	7	fassbender	michael	aguilar de nerha	3	10/01/17	11h00	Non	4
Kurzel	Justin	8	cotillard	marion	sofia	3	10/01/17	11h00	Non	4
Kurzel	Justin	9	irons	jeremy	rikkin	3	10/01/17	11h00	Non	4
Kurzel	Justin	10	gleeson	brendan	joseph lynch	3	10/01/17	11h00	Non	4
Kurzel	Justin	11	rampling	charlotte	Ellen Kaye	3	10/01/17	11h00	Non	4
Kurzel	Justin	12	labed	Ariane	Maria	3	10/01/17	11h00	Non	4
Chazelle	Damien	13	gosling	ryan	Sebastian	4	10/01/17	22h	Non	5
Chazelle	Damien	14	stone	emma	Mia	4	10/01/17	22h	Non	5
Chazelle	Damien	15	legend	john	Keith	4	10/01/17	22h	Non	5
Chazelle	Damien	16	DeWitt	Rosemarie	Laura	4	10/01/17	22h	Non	5
Chazelle	Damien	17	Wittrock	Finn	Greg	4	10/01/17	22h	Non	5
Chazelle	Damien	18	Simmons	J.K	Bill	4	10/01/17	22h	Non	5

FIGURE 7 – Troisième partie de notre table contenant tous les attributs et quelques tuples

nomC	prenomC	numRes	nbPlaceStandardRes	nbPlaceHandicapeRes	nbDboxRes
Martin	herve	4	2	0	0
Martin	herve	4	2	0	0
Martin	herve	4	2	0	0
Martin	herve	4	2	0	0
Martin	herve	4	2	0	0
Martin	herve	4	2	0	0
Dupont	Marc	8	2	1	0
Dupont	Marc	8	2	1	0
Dupont	Marc	8	2	1	0
Dupont	Marc	8	2	1	0
Dupont	Marc	8	2	1	0
Dupont	Marc	8	2	1	0
Dubois	Jeremie	15	2	0	2
Dubois	Jeremie	15	2	0	2
Dubois	Jeremie	15	2	0	2
Dubois	Jeremie	15	2	0	2
Dubois	Jeremie	15	2	0	2
Dubois	Jeremie	15	2	0	2
Morgan	philippe	16	1	0	2
Morgan	philippe	16	1	0	2
Morgan	philippe	16	1	0	2
Morgan	philippe	16	1	0	2
Morgan	philippe	16	1	0	2
Morgan	philippe	16	1	0	2
David	jean	23	0	0	2
David	jean	23	0	0	2
David	jean	23	0	0	2
David	jean	23	0	0	2
David	jean	23	0	0	2
David	jean	23	0	0	2

FIGURE 8 – Quatrième partie de notre table contenant tous les attributs et quelques tuples

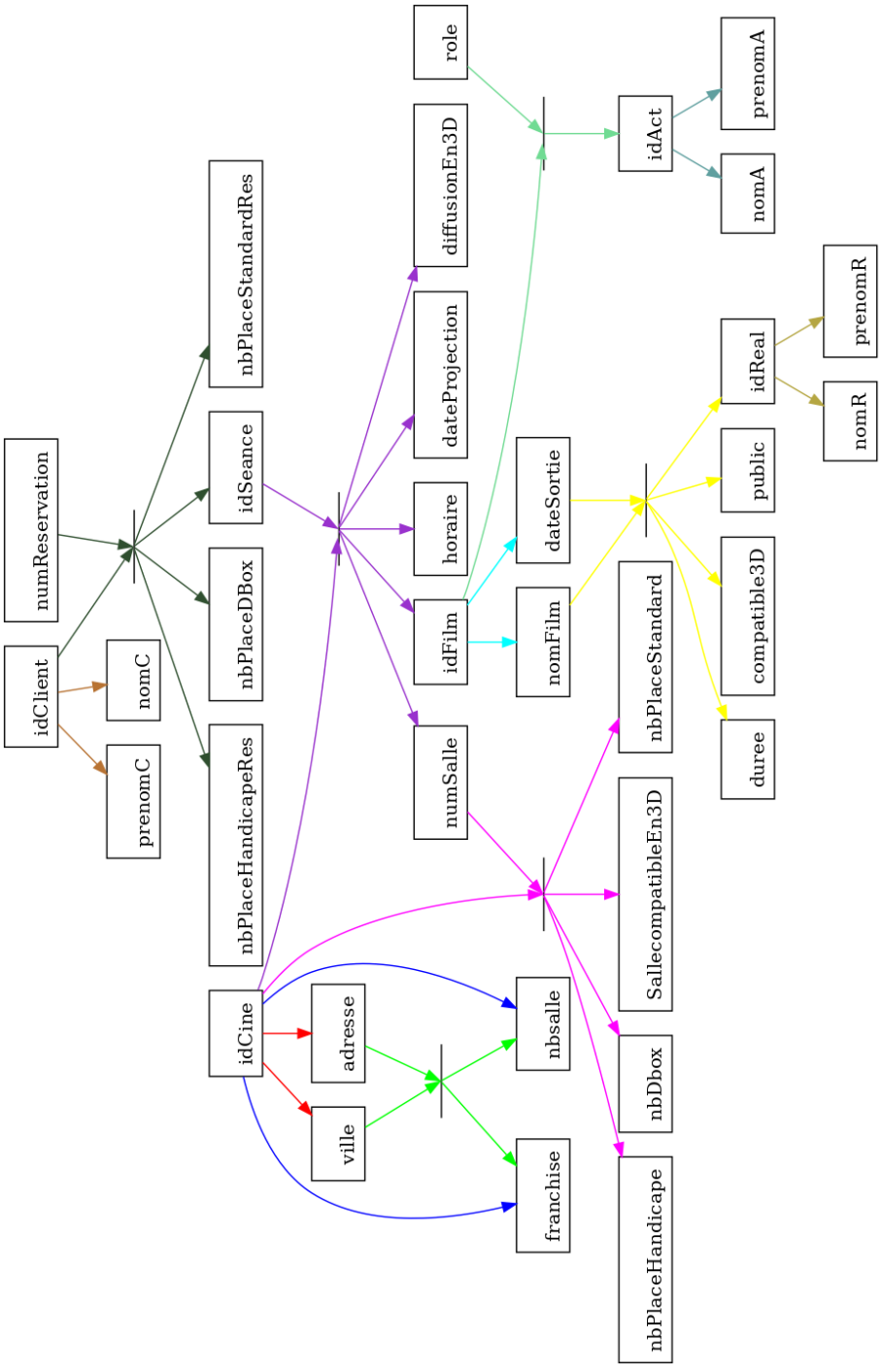

FIGURE 9 – Graphe des dépendances