Justification de notre modèle EA

Dans la liste des films notés, nous souhaitons pouvoir trier par moyenne des notes, par popularité (nombre de notes pour un film), par date de sortie. Il sera aussi possible de filtrer par date de sortie (avec un début et une fin), le nombre moyen d'étoiles (qui est notre manière de noter un film), par genre, par acteur et par mot présent dans le titre. Bien évidemment, un utilisateur ne pourra noter qu'une seule fois un film.

Ceci justifie donc que Noter soit une association, qui porte deux attributs : la note (entre 0 et 5 étoiles dans notre cas) et un commentaire (qui peut être null). Pour les cardinalités :

- un utilisateur peut noter une seule fois chaque film, mais autant de film qu'il souhaite (aucun, un ou plusieurs) ;
- un film est présent dans la base de données s'il a au moins une note. Il peut être noté par 1 ou plusieurs utilisateurs ;

Pour le film, nous n'avons donc pas besoin de stocker autre chose que une référence à ses données dans l'API, de cette manière on pourra afficher des détails sans les stocker dans notre base de données. Bien sûr, pour nos tris et filtres, nous avons besoin du titre et de la date de sortie, que l'on stocke donc.

Quand un film sera noté, si ça référence dans l'API correspondant à la clé primaire de notre table Film, est déjà présent, il suffira d'ajouter une note (pour l'utilisateur qui est loggué). Sinon, il faudra insérer ce nouveau film dans la base données, avec pour référence l'identifiant dans l'API. Il faudra aussi ajouter (s'ils n'existent pas déjà) les acteurs et le ou les genres de ce film dans les tables correspondantes. C'est pour ces raisons que nous utilisons comme clefs primaires les clefs de l'API et pas une autre clef, sans signification car alors il faudrait parcourir toutes les tables pour savoir si un film existe déjà dans la base de données.

Nous supposons qu'un film pourrait avoir aucun acteur (comme un dessin animé ou un documentaire) comme en avoir plusieurs. C'est pourquoi la cardinalité de Jouer de Film vers Acteur est de 0,N. Pour qu'un acteur existe dans la base de données, c'est qu'il a joué au moins dans un des films présents dans la table Films, d'où 1,N.

D'après nos recherches, tous les films ont au moins un genre, d'où la cardinalité de 1,N. Le même genre peut être donné à plusieurs films et de la même manière que pour un acteur, il faut au minimum un film de ce genre pour qu'il existe dans la base de données.