Description de la transformation du schéma Entité-Association (E/A) au modèle relationnel

1. Analyse du Schéma Entité-Association (E/A)

Le schéma Entité-Association est une représentation conceptuelle de la base de données. Dans cette étape, nous avons identifié les entités principales de notre système de gestion de co-living, ainsi que leurs relations. Les entités principales incluent **COMPLEXE**, **LOGEMENT**, **EQUIPEMENT**, **RESIDENT**, **RESERVATION**, **MAINTENANCE**, **EVENEMENT**, et **PARTICIPE**.

Les principales relations identifiées sont :

- Un **COMPLEXE** contient plusieurs **LOGEMENT** (relation 1:n).
- Un **LOGEMENT** peut avoir plusieurs **EQUIPEMENT** (relation 1:n).
- Un **RESIDENT** peut avoir plusieurs **RESERVATION** (relation 1:n).
- Une **RESERVATION** est associée à un seul **LOGEMENT** et un seul **RESIDENT** (relation 1:1).
- Un **EQUIPEMENT** peut avoir plusieurs **MAINTENANCE** (relation 1:n).
- Un **RESIDENT** peut participer à plusieurs **EVENEMENT** via la relation **PARTICIPE** (relation n:n).

2. Transformation des Entités en Tables

Les entités de notre schéma E/A sont directement transformées en **tables relationnelles** dans le modèle relationnel. Chaque entité devient une table, et chaque attribut de l'entité devient une colonne de la table correspondante.

Exemple :

- L'entité **COMPLEXE** devient la table **COMPLEXE** avec les colonnes `idComplexe`, `ville`, `codePostal`, `rue`, `numeroRue`.
- L'entité **LOGEMENT** devient la table **LOGEMENT** avec les colonnes `idLogement`, `type`, `nombreChambres`, `prix`, et une clé étrangère `idComplexe` qui lie chaque logement à un complexe spécifique.

3. Gestion des Relations et des Cardinalités

Les relations entre les entités, déterminées par les cardinalités, sont traduites en **clés étrangères (FK)** et parfois en **tables de jointure** pour gérer les relations n:n.

- **Relation 1:n (ex. COMPLEXE - LOGEMENT)** :
    - La table **LOGEMENT** contient une clé étrangère `idComplexe`, qui permet de lier chaque logement à un complexe. Cette relation est simplement implémentée par une clé étrangère dans la table **LOGEMENT**.
- **Relation 1:1 (ex. RESERVATION - LOGEMENT)** :
    - La table **RESERVATION** contient une clé étrangère `idLogement`, qui pointe vers le logement réservé. Cela représente la relation 1:1 entre une réservation et un logement spécifique. Chaque réservation ne peut concerner qu'un seul logement.
- **Relation 1:n (ex. LOGEMENT - EQUIPEMENT)** :
    - La table **EQUIPEMENT** contient une clé étrangère `idLogement`, permettant de lier un équipement à un logement particulier. Ainsi, un logement peut avoir plusieurs équipements.
- **Relation n:n (ex. RESIDENT - EVENEMENT via PARTICIPE)** :
    - Cette relation est plus complexe et nécessite une **table de jointure**, ici la table **PARTICIPE**. La table **PARTICIPE** lie un **RESIDENT** à un **EVENEMENT**, et contient les clés étrangères `idResident` et `idEvenement`. Cela permet de gérer le fait qu'un résident peut participer à plusieurs événements, et qu'un événement peut avoir plusieurs participants.

4. Application des Formes Normales (1NF, 2NF, 3NF)

Nous avons appliqué les règles des formes normales pour garantir l'intégrité des données et la réduction des anomalies de mise à jour.

1ère forme normale (1NF) :

- Chaque table contient des **valeurs atomiques**, c'est-à-dire que chaque colonne ne peut contenir qu'une seule valeur par enregistrement.
- Par exemple, la table **LOGEMENT** contient des colonnes telles que `type`, `nombreChambres`, et `prix`, toutes atomiques.

2ème forme normale (2NF) :

- Toutes les colonnes non-clés dépendent **entièrement de la clé primaire**.
- Par exemple, dans la table **RESERVATION**, les colonnes `dateEntree`, `dateSortie`, `idLogement`, et `idResident` dépendent directement de la clé primaire `idReservation`.

3ème forme normale (3NF) :

- Aucune colonne non-clé ne dépend d'une autre colonne non-clé.
- Par exemple, dans la table **EQUIPEMENT**, les colonnes `typeEquipement` ne dépendent que de la clé primaire `idEquipement`, et non d'une autre colonne.

5. Création des Tables de Jointure

Certaines relations **n:n** nécessitent des tables de jointure pour être représentées correctement dans le modèle relationnel.

- La relation entre **RESIDENT** et **EVENEMENT** est une relation n:n. Elle est implémentée par la table **PARTICIPE**, qui contient les clés étrangères `idResident` et `idEvenement`. Cette table permet de gérer les événements auxquels un résident participe.

6. Représentation Finale du Modèle Relationnel

Après avoir appliqué ces étapes, nous obtenons la représentation relationnelle suivante :

- **COMPLEXE** : Contient des informations sur les complexes immobiliers.
- **LOGEMENT** : Contient des informations sur les logements, avec une clé étrangère vers **COMPLEXE**.
- **EQUIPEMENT** : Contient des informations sur les équipements, avec une clé étrangère vers **LOGEMENT**.
- **RESIDENT** : Contient des informations sur les résidents.
- **RESERVATION** : Contient des informations sur les réservations, avec des clés étrangères vers **LOGEMENT** et **RESIDENT**.
- **MAINTENANCE** : Contient des informations sur les interventions de maintenance, avec une clé étrangère vers **EQUIPEMENT**.
- **EVENEMENT** : Contient des informations sur les événements communautaires.
- **PARTICIPE** : Table de jointure pour gérer la relation n:n entre **RESIDENT** et **EVENEMENT**.

Conclusion

La transformation du schéma Entité-Association en modèle relationnel repose sur l'application des concepts de cardinalité, de clés primaires et étrangères, ainsi que des règles de normalisation pour garantir une gestion cohérente des données. Chaque relation et chaque entité a été minutieusement traduite pour s'assurer que le modèle relationnel respecte les exigences du système de gestion de réseau de co-living tout en optimisant l'intégrité des données.