KRISSAAN Amen Allah, M2 TRIED

Rapport

Projet Base de Données – BDD

https://github.com/amenallah01/PROJET-BDD-JAZZ.git

I. Introduction :

Les bases de données sont au cœur des systèmes d'information modernes, offrant des solutions robustes pour stocker, organiser et manipuler les données de manière efficace· Ce projet s'inscrit dans cette démarche en proposant la conception et la mise en œuvre d'une base de données intitulée "Jazz & BD", spécifiquement dédiée à la gestion d'événements de concerts de jazz·

II. Présentation des données :

Les données exploitées dans ce projet sont réparties en plusieurs tables principales :

- Salles : Informations sur les lieux où se déroulent les concerts, incluant leur capacité·
- Organisateurs : Détails sur les entités responsables de l'organisation des concerts·
- Musiciens : Liste des musiciens participant aux concerts, ainsi que leurs rôles (leader, arrangeur, auteur)·
- Clients : Données des clients ayant réservé des billets pour les concerts·
- Morceaux : Catalogue des morceaux joués pendant les concerts, avec leurs auteurs et arrangeurs·

Ces tables, accompagnées de leurs relations, servent de base à la modélisation de la base de données et permettent de répondre à des requêtes liées à la gestion des concerts·

III. Objectifs :

L'objectif principal est de créer une structure de données conforme aux normes relationnelles, en respectant les trois formes normales, tout en développant des requêtes SQL adaptées à des cas pratiques· Ce projet permettra non seulement de répondre aux besoins opérationnels liés à l'organisation des concerts, mais également de démontrer une maîtrise approfondie des principes fondamentaux de gestion des bases de données·

On doit :

Modéliser et implémenter :

- Élaborer un modèle Entité-Association (E/A) pour représenter les relations entre les différentes entités·
- Transformer ce modèle conceptuel en un modèle relationnel opérationnel·

Répondre à des besoins pratiques :

- Rédiger et exécuter des requêtes SQL pour répondre à des questions spécifiques sur les concerts, les réservations et les musiciens·
- Fournir des analyses et statistiques à partir des données disponibles·

IV. Synthèse des Expérimentations :

① - Détermination des Cardinalités:

A ce stade la , pour déterminer les cardinalités dans notre problématique, nous avons minutieusement analysé toutes les précisions fournies dans l'énoncé du projet· En cas d'absence d'information explicite, nous avons opté pour des valeurs logiques afin d'estimer les cardinalités minimales et maximales· De plus, les données présentes à la fin de l'énoncé ont été utilisées pour vérifier et valider notre travail·

Le schéma Entité-Association (E/A) initial, préalablement défini par l'organisation et devant être respecté, est présenté ci-dessous :

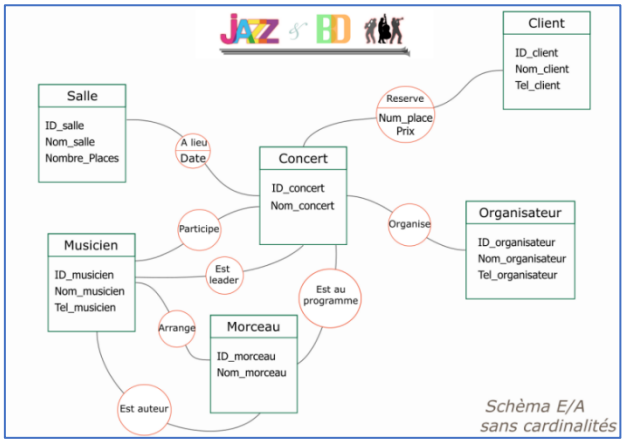

Figure 1- schéma : E/A

On Commence par :

1· Concert et Salle (A lieu) :

- Concert → Salle : Initialement, chaque concert devait se dérouler dans une seule salle « 1,1 »· Cependant, les données montrent qu'un concert peut avoir lieu dans plusieurs salles· Cardinalité : « 1,N »·
- Salle → Concert : Une salle peut accueillir plusieurs concerts ou ne pas en accueillir· Les données confirment que toutes les salles sont utilisées· Cardinalité finale : « 1,N »

2· Concert et Organisateur (Organise) :

- Concert → Organisateur : Chaque concert est organisé par un seul organisateur· Cardinalité : « 1,1 »·
- Organisateur → Concert : Un organisateur peut gérer plusieurs concerts, mais les données montrent que certains organisateurs n'en organisent aucun· Cardinalité finale : « 0,N »·

3· Concert et Musicien (Participe et Est leader) :

- Concert → Musicien (Participe) : Chaque concert doit avoir au moins un musicien, mais peut en inclure plusieurs· Cardinalité : « 1,N »·
- Musicien → Concert (Participe) : Un musicien peut participer à aucun, un ou plusieurs concerts· Cardinalité : « 0,N »·
- Concert → Musicien (Est leader) : Chaque concert a un leader unique· Cardinalité : « 1,1 »·
- Musicien → Concert (Est leader) : Un musicien peut ne pas être leader ou l'être pour plusieurs concerts· Cardinalité : « 0,N »·

4· Concert et Morceau (Est au programme) :

- Concert → Morceau : Chaque concert comprend au moins un morceau, mais peut en inclure plusieurs· Cardinalité : « 1,N »·
- Morceau → Concert : Un morceau peut ne pas être joué ou être programmé dans plusieurs concerts· Cardinalité : « 0,N »·

5· Concert et Client (Réserve) :

- Concert → Client : Chaque concert doit avoir au moins un billet vendu, mais plusieurs clients peuvent acheter des billets pour un même concert· Cardinalité : « 1,N »·
- Client → Concert : Chaque client doit avoir acheté au moins un billet, et il peut en acheter pour plusieurs concerts différents· Cardinalité : « 1,N »·

6· Morceau et Musicien (Est auteur) :

- Morceau → Musicien : Chaque morceau doit avoir au moins un auteur musicien, mais il peut être co-écrit par plusieurs musiciens· Cardinalité : « 1,N »·
- Musicien → Morceau : Un musicien peut ne pas être auteur (aucun morceau écrit) ou en avoir écrit plusieurs· Cardinalité : « 0,N »·

7· Morceau et Musicien (Arrange) :

- Morceau → Musicien : Chaque morceau doit être arrangé par au moins un musicien, mais il peut aussi être arrangé par plusieurs· Cardinalité : « 1,N »·
- Musicien → Morceau : Un musicien peut ne pas avoir arrangé de morceau ou avoir arrangé plusieurs morceaux différents· Cardinalité : « 0,N »·

⇨ Ces cardinalités reflètent la structure des entités et des associations définies dans le projet, tout en respectant les règles spécifiques au système "Jazz & BD"· Elles garantissent la cohérence et la logique du modèle conceptuel·

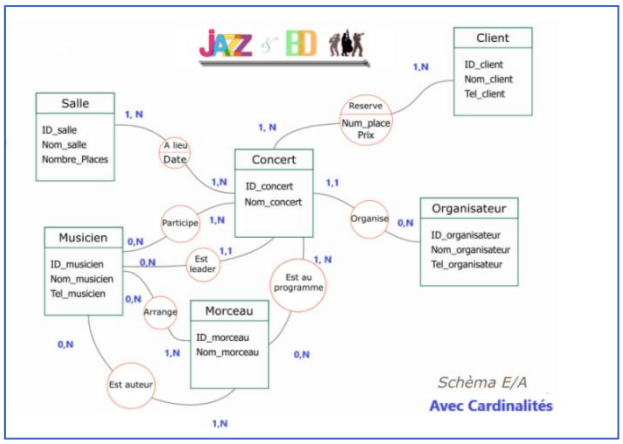

Figure 2 - E/A, Avec Cardinalités

② - Passage au Modèle Relationnel et Vérification de la conformité du modèle relationnel aux 3 formes normales.

Maintenant il est essentiel de respecter un ensemble de règles tout en s'appuyant sur les entités, les associations, et les cardinalités définies précédemment· Les principales règles à respecter sont les suivantes :

- ✓ Chaque entité est représentée sous forme de relation (table)·
- ✓ Les attributs des entités deviennent des colonnes de leurs relations respectives·
- ✓ La clé primaire d'une entité devient la clé primaire de la table correspondante·
- ✓ Une association binaire avec une cardinalité maximale de "N" des deux côtés génère une nouvelle relation (table)· Sa clé primaire est la concaténation des clés primaires des entités associées, et ces clés deviennent également des clés étrangères dans cette relation·
- ✓ Une association binaire avec une cardinalité maximale de "1" d'un côté génère une table avec les clés étrangères appropriées·

Création des premières tables :

- À partir des entités identifiées dans le schéma E/A, les relations initiales sont définies comme suit :
  - Salle : (ID salle, Nom salle, Nombre Places)
  - Organisateur: (ID organisateur, Nom organisateur, Tel organisateur)
  - Musicien : (ID musicien, Nom musicien, Tel musicien)
  - Morceau : (ID morceau, Nom morceau)
  - Client: (ID client, Nom client, Tel client)

Ces relations ont été dérivées directement des entités en suivant ces étapes :

- Chaque rectangle dans le schéma E/A correspond à une entité·
- Pour chaque entité, une table relationnelle est créée·
- Les attributs de l'entité deviennent les colonnes de la table correspondante·
- Les clés primaires, uniques et non nulles, sont représentées par les champs "ID"·

Création des tables pour les associations :

Pour les associations suivantes : "arrange", "est auteur", "réserve", "participe", et "est au programme", chaque association est binaire avec une cardinalité maximale de "N" des deux côtés· Par conséquent, elles génèrent des tables où :

- ✓ La clé primaire est la concaténation des clés primaires des entités associées·
- ✓ Les clés primaires des entités associées deviennent également des clés étrangères·

Les relations correspondantes sont définies comme suit :

- Arrange: (ID morceau, ID musicien)
- Est auteur: (ID morceau, ID musicien)
- Réserve: (ID concert, ID client, Num place, Prix)
- Participe: (ID concert, ID musicien)
- Est au programme: (ID concert, ID morceau)

En appliquant ces règles et méthodologies, nous obtenons un modèle relationnel robuste, conforme aux spécifications du projet et prêt à être exploité pour répondre aux besoins de gestion des concerts de jazz·

Pour assurer une gestion efficace des concerts et optimiser notre modèle relationnel, nous avons défini plusieurs relations clés :

- Réservation(#ID client, #ID concert, Num place, Prix)
- Auteur (#ID musicien, #ID morceau)
- Arrangement (#ID musicien, #ID morceau)
- Participation (#ID musicien, #ID concert)
- Programme (#ID morceau, #ID concert)

Toutefois, pour gérer les différentes séances d'un concert, nous introduisons une nouvelle relation :

Séance (#ID concert, Date, #ID salle)

Cette table permet de distinguer les occurrences d'un même concert jouées à différentes dates, tout en respectant la contrainte qu'un concert se déroule toujours dans la même salle· La clé primaire de cette relation est la combinaison « ID

concert, Date », assurant l'unicité de chaque séance· La référence à « ID salle » en tant que clé étrangère établit le lien avec le lieu de la représentation·

Concernant la table « Concert », sa structure devient :

Concert (ID concert, Nom concert, #ID musicien (leader), #ID organisateur)

Nous avons exclu « ID salle » de cette relation puisque nous avons introduit la table « Séance » pour gérer la multiplicité des représentations d'un concert·

Problèmes de Normalisation et Correction

En analysant ces relations sous l'angle des formes normales, nous constatons que notre modèle initial présente des redondances et des anomalies de mise à jour, ne respectant pas totalement les trois formes normales :

1· Première forme normale (1NF) :

Nos relations respectent cette forme, car elles possèdent toutes une clé primaire et ne contiennent que des valeurs atomiques·

2· Deuxième forme normale (2NF) :

Certaines relations ne sont pas conformes à la 2NF, car elles contiennent des « attributs non-clés dépendant uniquement d'une partie de la clé primaire composite »·

- -Exemple :
  - Dans « Réservation », l'attribut « Nom concert » dépend uniquement de « ID concert » et non de l'ensemble « ID concert, ID client », ce qui viole la 2NF·

3· Troisième forme normale (3NF) :

Comme certaines relations ne sont pas en 2NF, elles ne respectent pas non plus la 3NF·

Pour garantir la conformité à cette forme, chaque attribut non-clé doit dépendre uniquement de la clé primaire et non d'un autre attribut non-clé·

Correction du Modèle Relationnel Final

Pour respecter ces principes et obtenir un modèle relationnel optimisé, nous devons :

- « Éliminer les dépendances partielles » en normalisant les relations·
- « Créer des tables supplémentaires si nécessaire » pour séparer les dépendances fonctionnelles·

En appliquant ces corrections, nous garantissons un « modèle relationnel cohérent, optimisé et conforme aux trois formes normales », évitant ainsi les redondances et assurant une gestion efficace des concerts de jazz·

Validation du Modèle Relationnel et Conformité aux Formes Normales

Afin d'assurer une structure optimale et conforme aux principes de normalisation, nous avons appliqué les trois formes normales (1NF, 2NF, 3NF) à notre modèle relationnel·

Les relations finales obtenues sont :

- Salle (ID salle, Nom salle, Nombre Places)
- Organisateur (ID organisateur, Nom organisateur, Tel organisateur)
- Musicien(ID musicien, Nom musicien, Tel musicien)
- Morceau (ID morceau, Nom morceau)
- Client (ID client, Nom client, Tel client)
- Concert (ID concert, Nom concert, #ID musicien (leader), #ID organisateur)
- Réservation (#ID Salle, #ID concert, Date, Num place, ID Client, Prix)
- Participation (#ID musicien, #ID concert)
- Programme (#ID morceau, #ID concert, #ID musicien)
- Auteur (#ID musicien, #ID morceau)
- Séance (#ID Salle, #ID concert, Date)

Optimisation du Modèle

Pour éviter une multiplication excessive des relations, nous avons combiné les tables « Programme » et « Arrangement » en une seule relation « Programme(#ID morceau, #ID concert, #ID musicien) »· Cette fusion réduit la redondance et améliore l'efficacité du modèle·

Conformité aux Trois Formes Normales

1· Première Forme Normale (1NF)

- Chaque table possède une clé primaire unique·
- Tous les attributs contiennent des valeurs atomiques (aucune liste ou ensemble imbriqué)·
  - ✓ 1NF validée

2· Deuxième Forme Normale (2NF)

- Toutes les relations en 1NF sont vérifiées·
- Chaque attribut non clé dépend de la totalité de la clé primaire et non d'une partie de celle-ci·
  - ✓ 2NF validée

3· Troisième Forme Normale (3NF)

- Chaque relation en 2NF est vérifiée·
- Aucun attribut non clé ne dépend d'un autre attribut non clé, éliminant ainsi toute dépendance transitive·
  - ✓ 3NF validée

Maintenant, notre modèle relationnel final est conforme aux trois formes normales, garantissant une organisation optimale et une gestion efficace des concerts de jazz·

③ - Contraintes d'Intégrité :

Application des Contraintes d'Intégrité et Vérification des Données

Pour garantir la cohérence et éviter les anomalies dans la base de données, il est essentiel d'appliquer des « contraintes d'intégrité » sur les données et relations· En s'appuyant sur le « modèle Entité-Association (E/A) » et les précisions fournies, voici les principales contraintes à respecter :

Contraintes d'Intégrité

Unicité des identifiants :

- « ID concert », « ID salle », « ID organisateur », « ID musicien », « ID morceau » et « ID client » doivent être uniques (clés primaires) ·

Contraintes sur les relations :

- Chaque concert doit avoir un seul leader, donc une « contrainte d'unicité » est imposée sur « ID concert, ID musicien » pour le rôle de leader·
- La combinaison « ID salle, Date » doit être unique pour garantir qu'une salle n'accueille pas plusieurs séances simultanées·
- « ID concert, Date » constitue une clé primaire composite pour assurer l'unicité de chaque séance·
- La combinaison « ID client, ID concert » doit être unique pour éviter qu'un même client ait plusieurs réservations pour un même concert·
- La combinaison « ID musicien, ID concert » doit être unique pour éviter qu'un musicien soit enregistré plusieurs fois dans un même concert·
- Chaque place réservée doit être unique pour une séance, garantissant qu'une place ne soit vendue qu'une seule fois·

Contraintes sur les valeurs:

- « NombrePlaces » doit être strictement positif·
- « Prix » doit être positif et peut dépendre de la salle et de la place attribuée·

Vérification et Correction des Données

Avant d'exécuter les requêtes SQL, nous avons vérifié l'intégrité des données :

- « Concert 3 »: Initialement répertorié dans deux salles différentes à des dates distinctes, ce qui a confirmé la cardinalité « 1,N » entre Concert et Salle·

- « Erreur dans les réservations »: Deux clients (« Annie » et « John ») avaient la même place pour le « Concert 2 »· Pour corriger cela, « John » a été déplacé de la place « 9 » à la place « 10 »·

Vérification des références :

- Chaque concert fait référence à une salle existante·
- Tous les concerts ont un organisateur valide·
- Les musiciens répertoriés existent bien dans la base·
- Chaque morceau joué est présent dans la liste des morceaux et les auteurs/arrangeurs sont bien référencés·

Ces Contraintes et corrections assurent un modèle relationnel fiable et conforme, évitant les incohérences et garantissant l'intégrité des données·

④ - Construire les requêtes SQL :

Mise en Place et Préparation de la Base de Données

En partant du modèle Entité-Association , nous avons créé et préparé notre base de données afin d'extraire les informations nécessaires à la gestion des concerts· Cette préparation inclut plusieurs étapes essentielles, décrites ci-dessous :

1- Connexion et Initialisation

- Connexion à la base de données via « sqlite3·connect('jazz_projet·db') », créant ainsi un objet de connexion·
- Création d'un « curseur SQL » permettant d'exécuter les commandes sur la base·

2- Chargement du Schéma de la Base de Données

- Lecture du fichier « DATA·sql » contenant la structure des tables·
- Découpage des requêtes SQL en utilisant « ; » comme séparateur·
- Exécution des commandes SQL pour créer les tables·
- Gestion des erreurs « OperationalError » avec un message « Command skipped » pour identifier les commandes non exécutées·

3- Affichage des Tables

- Itération sur la liste des tables pour afficher les colonnes et les premières lignes de chaque table·

4-Insertion des Données

- Lecture des fichiers pré-formatés « xJazz_clients_all·txt », etc·
- Utilisation de « INSERT INTO » pour insérer les données et validation après chaque insertion « commit() »·
- Pour la table « Auteur », récupération préalable de l' « ID du musicien » avant l'insertion·
- Gestion des erreurs d'intégrité et affichage des messages d'erreur·

5 - Insertion des Données des Concerts

- Lecture des concerts à partir du fichier « donnees_concerts·json »·
- Insertion des données dans les tables « Concert, Programme, Participation et Séance » en utilisant des sous-requêtes pour récupérer les « ID correspondants »·
- Insertion des « réservations » dans la table « Reservation » pour chaque séance·

6 - Affichage des Résultats

- Création d'une fonction « print_requete » qui exécute une requête SQL et affiche les résultats sous forme tabulaire pour une meilleure lisibilité·

Construction des Requêtes SQL

Après la mise en place de la base, nous avons rédigé les requêtes SQL nécessaires pour répondre aux questions posées sur les concerts· Dans la plupart des cas, plusieurs approches ont été testées· Cependant, dans ce rapport, une seule méthode sera détaillée, tandis que les autres alternatives seront disponibles dans le notebook joint·

Grâce à cette préparation rigoureuse, notre base de données est prête à être interrogée de manière efficace pour la gestion des concerts de jazz·

Conclusion

Le projet "Jazz BD" a démontré l'importance d'une base de données bien conçue et structurée pour la gestion efficace des concerts de jazz·

Grâce à une modélisation rigoureuse t au respect des principes de normalisation, nous avons mis en place un système robuste capable de gérer avec précision les informations liées aux événements musicaux·

Ce projet illustre ainsi le rôle fondamental des bases de données dans la gestion d'événements, en offrant un outil performant pour l'analyse, la planification et la prise de décisions stratégiques·