Modélisation

Peng Cédric, Lin Michel

May 18, 2025

1 Présentation

Nous avons décidé de faire un système de réservation d'entré et de billet pour un festival de restaurants séparé en plusieurs zones où les utilisateurs doivent passer par une préinscription pour participer au festivals pour une période précise, puis d'une sélection et achat de leur billet d'entré avec créneaux d'achat.
De nouvelle sélection sera faite apres la fin d'une selection si il reste des places dans le festival et que le festival n'a pas encore démarré.
L'utilisateur devra payer le prix du billet qui dépend de plusieurs facteurs(zone, statut, pass, ...) et acheter des "crédits" qui feront office de monnaie dans les restaurants. Un utilisateur ne peut avoir qu'un seul billet par période de festival.

Une fois dans le festival, le billet fera office de carte où sera stocké les crédits acheté précédemment. Le billet sera alors nécessaire pour consommer ce qui permettra ainsi de conserver un historique des consommation et activité fait par un utilisateur.

Nos restaurants possèdent chacun un menu avec des plats et des prix selon la zone où elle se situe. La zone 1 correspond au restaurant dont les prix des plats ne dépasse pas 50 crédits, la zone 2 au restaurants avec des prix ne dépassant pas 200 crédits, et enfin la zone 3 ou les restaurants seront qualifié de gastronomique avec des prix pouvant aller au delà de 200 crédits. Il sera possible de consulter l'addition détaillé de chaque consommation dans un restaurants avec les plats, les quantités prise et la note total.

2 Modèle E\R

2.1 Réservation

- Une phase de préinscription pour pouvoir être sélectionné par la suite.

- **Check : on peut participer a la préselection que 2 semaine avant le début du festival**

- Une phase de sélection qui permet de choisir les utilisateurs qui seront choisit pour l'achat du billet.
  Trigger : Un utilisateur ne peut pas faire la sélection si il n'a pas fait la préinscription
  trigger : Les selections et les préinscriptions se déroule que pendant l'intervalle de temps autorisé $Selection(id\_u, id\_festival) \subset Preinscription(id\_u, id\_festival)$
- Une fois sélectionné l'utilisateur aura une journée précis pour acheter son billet.
  Trigger : L'achat d'un billet ne peut se faire que si l'utilisateur à fait la préinscription, passé la sélection et durant la journée fixé.
  $Utilisateur\_Billet(id\_u, id\_festival) \subset Selection(id\_u, id\_festival)$
  $CURRENT\_TIME \in Selection(creneau)$
- Si des places sont encore disponible on peut relancer une phase de sélection jusqu'a qu'il n'en reste plus ou que le festival commence.
- Les personnes qui ont été sélectionnées et qui n'ont pas procédé à un achat peuvent à nouveau l'être.
  Trigger : La sélection est faite avec les personnes qui se sont préinscrite mais qui n'avait pas été sélectionné ou fait d'achat
- Un tarif selon le statut de l'utilisateur (retraité,etc..)
  Fonction : Vérifier le status de l'utilisateur lors de l'achat
- Possibilité d'acheter un pass pour avoir des avantages.
- Un tarif selon les zones d'accès prise. Un billet ne donne par défaut pas accès à toute les zones du festival. L'utilisateur doit préciser à quelle zone il souhaite avoir accès. Un coût supplémentaire sera ajouté selon les zones choisies.
  Fonction : Le prix du ticket d'entré correspond à la somme des différents tarifs (statut,période, bonus pass,...)
- Un système de points de fidélité en fonction du nombre de crédit acheté
  Trigger : Les points de fidélité sont réinitialisés si l'utilisateur n'est pas venu depuis plus d'un an
- Possibilité de tracer tout les achats effectués par un utilisateur
- Le nombre de billet disponible pour une période est défini a l'avance
- La phase de sélection continue jusqu'à la date de début du festival.
  Trigger : Il doit resté des places disponibles
- **Contrainte d'unicité : On ne peut acheter que un seul billet pour une période donnée**
- **Trigger : On ne peut pas acheter de billet si le festival a déjà commencé.**

2.2 Pass

- Plusieurs sorte de nom/type de bonus (cuillère de bronze, couteau de fer, fourchette d'or) qui chacun offre différentes réduction sur le prix du billet d'entré ainsi qu'un bonus de crédit inital .
- **Fonction : Un utilisateur ne peut pas avoir plusieurs pass actif, seul celui avec la date d'achat la plus récente est pris en compte**
- Un prix selon le type et de la durée du pass
  Check : Date = 1mois — 6mois — 1an
- Ajout d'un bonus de crédit initial si l'utilisateur possède un pass **Trigger : Le crédit initial dépend du pass**
- réduction du prix du billet d'entré selon le type de pass **Trigger : Suivant le pass que l'utilisateur possede**
- un pass est composé d'une durée et d'un nom/type de bonus.
- le nom/type du bonus détermine la réduction et le bonus de crédit initial
- plusieurs utilisateurs peut avoir le meme type de pass

2.3 Restaurant

- Les restaurants seront chacun situé dans une zone qui détermine leur type (chère, moyen, très chère)
  Contrainte d'unicité : Un restaurant ne peut pas etre dans plusieurs zones en meme temps Trigger : Le billet de l'utilisateur doit donner accès a la zone du restaurant pour y consommer
- Les restaurants possèdent un menu avec différents plats et prix
  Contrainte d'unicité : Il ne peut pas avoir le meme plat avec un prix different
- Chaque plat ne peut être vendu qu'un nombre limité de fois par jour
  Trigger : On ne peut pas commandé le plat en question s'il n'est plus en stock
- On ne peut pas consommer un plat qui n'est pas dans le menu du restaurant
  $Recapitulatif(num\_zone\#, nom\_restaurant\#, nom\_plat) \subset Menu(num\_zone\#, nom\_restaurant\#, nom$
- **Trigger : La présence d'un billet valide est nécessaire pour consommer dans un restaurant**
- L'utilisateur à accès a l'historique des additions payées. Celle ci contiendra les plats et la quantité commandé ainsi que le montant total.

- un Restaurant ne peut proposé qu'au plus 10 plat dans son menu
  Trigger : verification qu'on ne dépasse pas le nombre de plat maximal dans un menu lors d'un update/insert

- Les restaurants ont des prix maximals selon la zone dans laquel elle se trouve
  Trigger : un restaurant ne pourra pas ajouter de plat a son menu violant cette règle

2.4 Schéma Relationnel

- Utilisateur(id_u, nom,prénom,est_etudiant,est_retraité)
  - id_u → nom, prénom,est_etudiant,est_retraité
- Festivals(id_festival, date_debut_festival, date_fin_festival, date_debut_preinscription,date_fin_preinscriptio
  date_debut_selection,date_fin_selection, prix_entree)
  - id_festival → date_debut, date_fin, prix_entree
- Billet(id_b, id_festival)
- Achat_Billet(id_b#, id_festival#, id_u#, prix_payé, credit_total)
  - id_b, id_festival,id_u → prix_payé,credit_total
- Abonnement(type_bonus#, durée, id_u#, date_d'achat)
- Type_Pass(type_bonus#, durée, prix)
  - type_bonus, prix → durée
  - type_bonus, durée → prix
- Bonus(type_bonus, réduction_billet, crédit_bonus)
  - type_bonus → réduction_billet, crédit_bonus
- Préinscription(id_u#, id_festival#)
- Sélection(id_u#, id_période#, créneau)
  - id_u, id_période → créneau
- Consommation(id_b#,num_zone,nom_restaurant#, date_horaire)
- Recapitulatif(id_b#,num_zone,nom_restaurant#, date_horaire, nom_plat#, quantité)
  - id_b, num_zone, nom_restaurant, date_horaire, nom_plat → quantité
- Menu(num_zone#, nom_restaurant##,nom_plat, quantité_max_quotidien, prix)

  - num_zone#, nom_restaurant#, nom_plat → quantité_max_quotidien, prix

- Restaurant(num_zon#, nom_restaurant, num_zone#,nom)
  - num_zone, nom_restaurant → num_zone, nom
- Participation(id_festival#,num_zone#, nom_restaurant##)
- Zone(num_zone, tarif_max_plat, tarif_entrée)
  - num_zone → tarif_max_plat, tarif_entrée
- Billet_Accès(id_b,id_festival#, num_zone#)

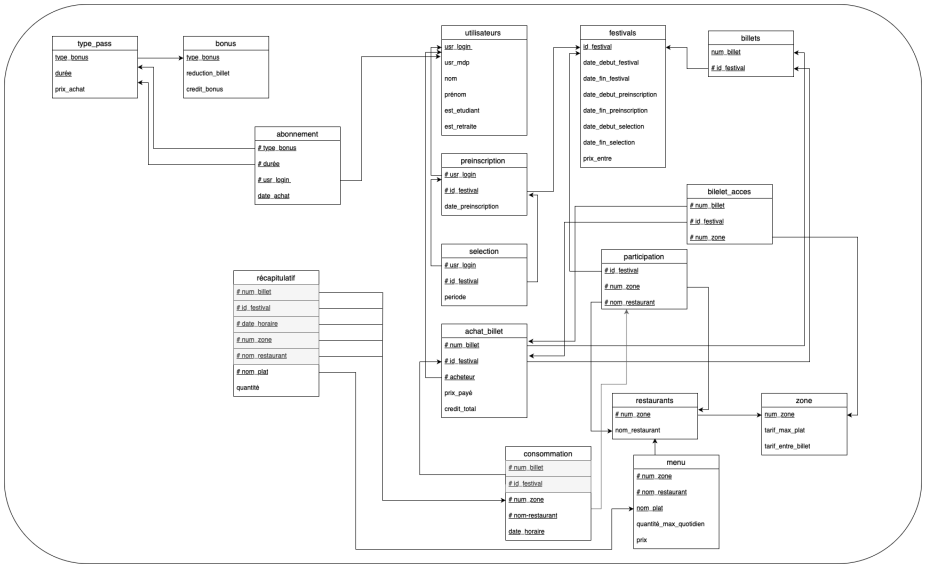

Figure 1: Schema relationnel.

3 Index

Dans notre base de données, nous utilisons principalement des index primaires, car nos clés primaires couvenrt déjà une grand partie des attributs de nos tables. Cela limite naturellement le besoin en index secondaire, tout en garantissant de bonne performances sur la majorité de nos requêtes. Toutefois, afin d'optimiser certains cas d'usage spécifiques, nous avons ajouté un index secondaire ciblé pour améliorer l'efficacité de certaines recherches précices comme la recherche d'intervalle de prix.

4 Transactions

On retrouve plusieurs utilisations de transactions dans notre systeme comme pour l'achat du billet ou de la création du ticket de consommation. (voir les scénarios) L'achat du billet utilise notamment un lock en lecture. La gestion d'erreur lors de la création d'un ticket de consommation utilise le rollack.

5 Scenarios

Dans le dossier ../ETUDES vous retrouverez plusieurs fichier .sql qui contienne des scénarios pour notre base de données.

5.1 creation_festivals.sql

Ce fichier SQL sert à simuler l'ajout d'un nouveau festival dans la base de données ainsi que l'ensemble du processus lié aux préinscriptions, sélections et achats de billets.

5.2 creation_consommation.sql

Ce fichier SQL permet de tester la création de consommations pendant un festival, la gestion concurrente des achats de plats par les participants, ainsi que le calcul du chiffre d'affaires des restaurants et du festival. Il vérifie également les règles métiers liées aux soldes et aux limitations de consommation.

5.3 creation_transaction.sql

Ce fichier SQL sert a tester les transactions. Celui-ci possède la meme base que creation_festivals.sql avec un exemple d'achat de billet qui est une action pouvant etre critique a l'aide des commandes SQL begin,commit,rollback. Les transactions sont mis en commentaire et doivent etre éxécuté a part dans 2 terminals après que le fichier soit exécuté.

5.4 creation_ticket_recapitulatif.sql

Ce fichier SQL sert a montrer un exemple d'utilisation d'une transaction avec rollback pour la gestion des erreurs lors de la creation d'un ticket de consommation lors de l'ajout d'un plat dans la consommation.

(voir ./ETUDES/readme.md pour plus d'informations)