Une image contenant Police, capture d’écran, Graphique, symbole

Description générée automatiquement

**Projet final**

**Gestion d’un casino**

Par Tarik Benakezouh, Carlos Eduardo Calle Guerrero,

Antoine Ho, Lucky Khounvongsa, & Kha Pham

Matricules : 20184524, 20220633, 20202790, 20172476 & 2018233

Travail présenté à Jihene Rezgui

dans le cadre du cours de

IFT2935-B – Base de données

**Remis le 10 avril 2024**

**Introduction**

Dans le monde dynamique et en constante évolution des jeux et divertissements, la gestion efficace d'un casino représente un défi majeur à entreprendre. Avec l'avancement technologique et la croissance des attentes des clients, il est impératif pour les casinos de moderniser leurs systèmes de gestion pour améliorer l'efficacité, garantir la sécurité et maximiser les profits. Ce projet vise à développer une solution de gestion de base de données sophistiquée pour un casino, couvrant tous les aspects cruciaux de son fonctionnement, y compris la gestion des installations, du personnel, des recettes des machines de jeu, et des interventions techniques. En utilisant SQL Server 2019, cette solution permettra au casino de gérer ses opérations de manière plus efficace et précise, en assurant une expérience de jeu fluide pour les clients et un environnement de travail optimisé pour le personnel.

**Description du projet**

**Analyse approfondie des besoins**

Le fonctionnement d'un casino moderne repose sur une gestion optimisée et automatisée de ses multiples composantes, allant des installations physiques aux ressources humaines, tout en passant par le suivi financier des activités de jeu. Une base de données conçue pour un casino doit, par conséquent, offrir une solution intégrée qui répond à ces exigences complexes. Ce projet vise à concevoir une base de données qui non seulement répond à ces besoins opérationnels, mais le fait de manière qui maximise l'efficacité, assure la conformité réglementaire, et optimise l'expérience des utilisateurs finaux, tant pour le personnel que pour les clients.

**Réalisme et adéquation**

Le réalisme de la conception de la base de données est assuré par l'identification précise et la modélisation des entités et des relations clés qui reflètent fidèlement les opérations d'un casino réel. Cela comprend :

* Les **installations**, qui sont diversifiées et comprennent des équipements à la fois statiques et dynamiques, nécessitant une gestion et un suivi détaillés pour assurer leur fonctionnement optimal et leur conformité aux normes de sécurité.
* La **gestion du personnel**, qui doit être adaptable pour couvrir divers rôles et compétences, tout en fournissant des informations essentielles pour la gestion des ressources humaines et la planification du personnel.
* Le **suivi des recettes**, essentiel pour la viabilité financière du casino, doit être précis et fiable, permettant une analyse détaillée des performances des différentes machines et installations.
* La **gestion des pannes** nécessitant une approche systématique pour minimiser les temps d'arrêt et maintenir une expérience client de haute qualité.

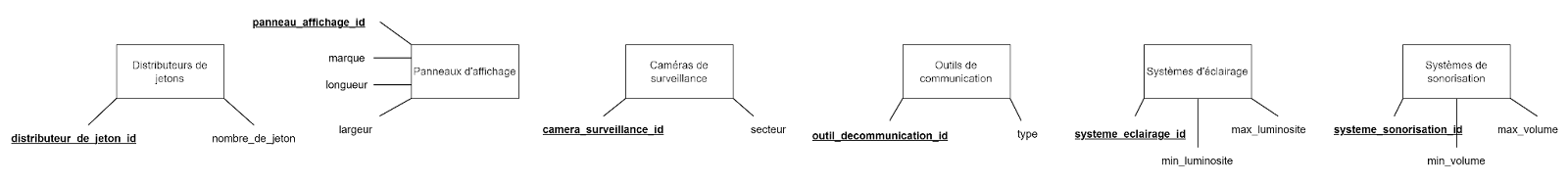
**Justification des choix de conception**

La conception du schéma Entité-Association (E-A) présentée repose sur une structuration minutieuse visant à optimiser la gestion et l'organisation des données au sein d'un casino. La décision de fusionner les trois catégories de personnel en une seule table "Personnel" tout en associant cette table à une autre dédiée aux rôles des personnels reflète une approche rationalisée. Cela simplifie la base de données en réduisant la redondance, tout en conservant la flexibilité nécessaire pour détailler les fonctions et responsabilités à travers la table des rôles, facilitant ainsi les requêtes et les mises à jour.

En ce qui concerne les installations, la décision de créer une table distincte pour chaque type d'installation (machines de jeu, distributeurs de jetons, panneaux d'affichage, etc.) est justifiée par la diversité et la complexité spécifique de chaque élément. Cette approche permet de gérer plus efficacement les caractéristiques uniques et les exigences de maintenance de chaque type d'installation, offrant une granularité fine dans le suivi et la gestion des ressources.

L'aspect des pannes souligne l'importance de la normalisation des données, en particulier le respect de la première forme normale (1FN). La création d'une table séparée "Type de panne" est une réponse stratégique à la nécessité de gérer des informations multiples et variées liées aux pannes, évitant ainsi les données non atomiques et assurant une meilleure intégrité des données. Cette séparation facilite les analyses détaillées des pannes et améliore la maintenance préventive et corrective.

L'inclusion d'une table "Pièce" sans référence explicite dans l'énoncé initial peut être vue comme une anticipation des besoins futurs ou une extension logique du modèle, permettant de suivre les pièces de rechange, leur état, et leur association avec les équipements et les interventions de maintenance. Bien qu'elle n'ait pas été spécifiée initialement, sa présence peut enrichir le schéma de données en offrant une vue complète sur la gestion des inventaires et des ressources matérielles, ce qui est crucial pour un fonctionnement optimal du casino.

Une image contenant capture d’écran, noir, Symétrie, noir et blanc

Description générée automatiquement**Schéma Entité-Association**

**Schéma relationnel**

MachineDeJeu (**machine\_jeu\_id**, date\_service, **#id\_jeu**)

Jeu(**id\_jeu**, nom\_jeu, min\_mise, max\_mise)

DistributeurDeJeton (**distributeur\_de\_jeton\_id**, nombre\_de\_jeton)

PanneauAffichage (**panneau\_affichage\_id**, marque, longueur, largeur)

CameraSurveillance (**camera\_surveillance\_id**, secteur)

OutilsDeCommunication (**outils\_de\_communication\_id**, type)

SystemeEclairage (**systeme\_eclairage\_id**, min\_luminosite, max\_luminosite)

SystemeSonorisation (**systeme\_sonorisation\_id**, min\_volume, max\_volume)

Adresse\_civique(**numero\_adresse, rue, ville**)

Role\_personnel(**id\_role**, nom, secteur, expertise, droit\_acces)

Personnel(**numero\_id**, nom, prenom, **#id\_role**, **#numero\_adresse**, **#rue**, **#ville**)

Recette(**recette\_id**, date, montant profit, coûts, **#machine\_jeu\_id**, rentable)

Panne(**panne\_id**, type, date, **#personnel\_id**, piece, **#id\_type\_panne**)

Type\_panne(**id\_type\_panne**, nom. catégorie)

Piece(**id\_piece**, nom, date\_service, etat, **#id\_fournisseur**, **#id\_machine\_jeu**, **#id\_panne**)

Fournisseur(**id\_fournisseur**, nom, ville)

**Normalisation**

Machine de jeu: (id) -> (date\_service, #id\_jeu)

Jeu: (id\_jeu) -> (nom\_jeu, min\_mise, max\_mise)

Distributeur de jeton: (id) -> (nombre de jeton, date\_service)

Panneau d'affichage: (id) -> (marque, date\_service, longueur, largeur)

Caméra de surveillance: (id) -> (secteur, date\_service)

Outil de communication: (id) -> (date\_service, type)

Système d'éclairage: (id) -> (date\_service, nom, min\_luminosite, max\_luminosite)

Système de sonorisation: (id) -> (date\_service, nom, min\_volume, max\_volume)

Adresse civique: (**numero\_adresse, rue, ville**) -> (numero\_adresse, rue, ville)

Rôle du personnel: (**id\_role**) -> (nom, secteur, expertise, droit\_acces)

Personnel: (**numero\_id**) -> (nom, prenom, #id\_role, #code\_postal, #numero\_adresse, rue, ville)

Recette: (**recette\_id**) -> (date, montant profit, coûts, #id\_machine\_jeu, rentable)

Panne: (**panne\_id**) -> (**#id\_type\_panne**, date, #id\_personel)

Type de panne: (**id\_type\_panne**) -> (nom, categorie)

Pièce: (**id\_piece**) -> (nom, date\_service, etat, #id\_fournisseur, #id\_machine\_jeu, #id\_panne)

Fournisseur: (**id\_fournisseur**) -> (nom, ville)

**Première forme normale (1FN)**

Passer à la première forme normale est pour nous une démarche assez intuitive puisqu’elle vise à garantir que chaque attribut contient des valeurs atomiques. Cette étape est essentielle pour éliminer les groupes de répétition et assurer que chaque champ stocke une valeur unique et spécifique. Cependant, nous avions initialement envisagé de regrouper des informations telles que le numéro d'adresse, la rue, et la ville en un seul attribut sous forme de concaténation. En plus du non-respect de la première forme normale, cela compliquait les requêtes nécessitant l'accès individuel à ces informations, comme pour trouver toutes les adresses dans une ville donnée. Nous avons ainsi résolu ce problème en séparant ces données en attributs distincts et atomiques, améliorant l'accessibilité et la manipulation des données.

**Deuxième forme normale (2FN)**

La deuxième forme normale se concentre sur l'élimination des dépendances partielles en veillant à ce que chaque attribut non-clé dépende entièrement de la clé primaire dans les situations où les clés composées sont présentes. Dans notre cas, l'application de la 2FN ne se manifeste pas de manière directement pertinente, principalement à cause de l'absence de clés composées dans la plupart des tables que nous avons définies. Chaque table semble avoir une clé primaire unique, indiquant que toutes les dépendances sont pleinement fonctionnelles par définition. Cette simplification de la structure nécessite cependant que nous soyons particulièrement attentifs lors de la définition de chaque table pour assurer l'unicité et l'adéquation de la clé primaire.

**Troisième forme normale (3FN)**

La troisième forme normale vise à éliminer les dépendances transitives, assurant que chaque attribut non-clé dépende directement de la clé primaire et non d'un autre attribut non-clé. Nous avons identifié que des tables telles qu’“Adresse civique”, “Rôle du personnel”, “Type de panne”, “Fournisseur”, et “Jeu” ne respectaient pas pleinement ce niveau de normalisation. Par exemple, les attributs d'un rôle peuvent ne pas dépendre directement de l'employé lui-même, mais sont liés par une clé étrangère. Afin d’y remédier, nous avons opté pour la création de nouvelles tables pour ces attributs interdépendants, avec une clé primaire distincte (souvent reprenant la clé étrangère de l'attribut dans la table originale), nous permettant de clarifier et de structurer ces relations de manière plus appropriée. Cette méthode facilite la maintenance des données, rend les mises à jour plus efficaces et améliore la cohérence de l'information en assurant que toutes les dépendances sont directement liées à la clé primaire appropriée.