UNIVERSITÉ LIBRE DE BRUXELLES

ULB

Faculté de Lettres, Traduction et Communication

Conception et gestion de banques de données : Rapport de projet

Budget Squirrel : Application de gestion de budget

Milena ALBU
Marie HUYSMAN

Rapport de projet écrit dans le cadre du cours STIC-B505 : Conception et gestion de banques de données

Année académique 2020–2021

Table des matières

1 Introduction

Si besoin de citer : (**SOURCE**) Notes de bas de page [1]

Dans le cadre de ce projet, nous avons choisi de développer une application permettant la gestion de budget à une échelle personnelle. Après une brève présentation du domaine d'application, nous présenterons le schéma conceptuel, ainsi que sa traduction en schéma relationnel. Ensuite, nous décrirons les différents aspects des scripts SQL que nous avons mis en place [...] . Enfin, nous exposerons brièvement l'application Web que nous avons développée. Vous trouverez, en annexe à ce rapport, deux scripts sql : `initdb.sql`, qui contient tous les scripts permettant la création de la base de donnée en elle-même, et `data.sql`, qui peuple la base de données de quelques informations, permettant ainsi de parcourir les différentes pages de l'application.

1.1 Domaine d'application

Utilisateurs cibles, cas d'utilisation du système

1. informations suppl.

2 Schéma conceptuel

Schéma conceptuel documenté, accompagné d'une description et justification des choix

! vérifier si tout correspond avec la DB

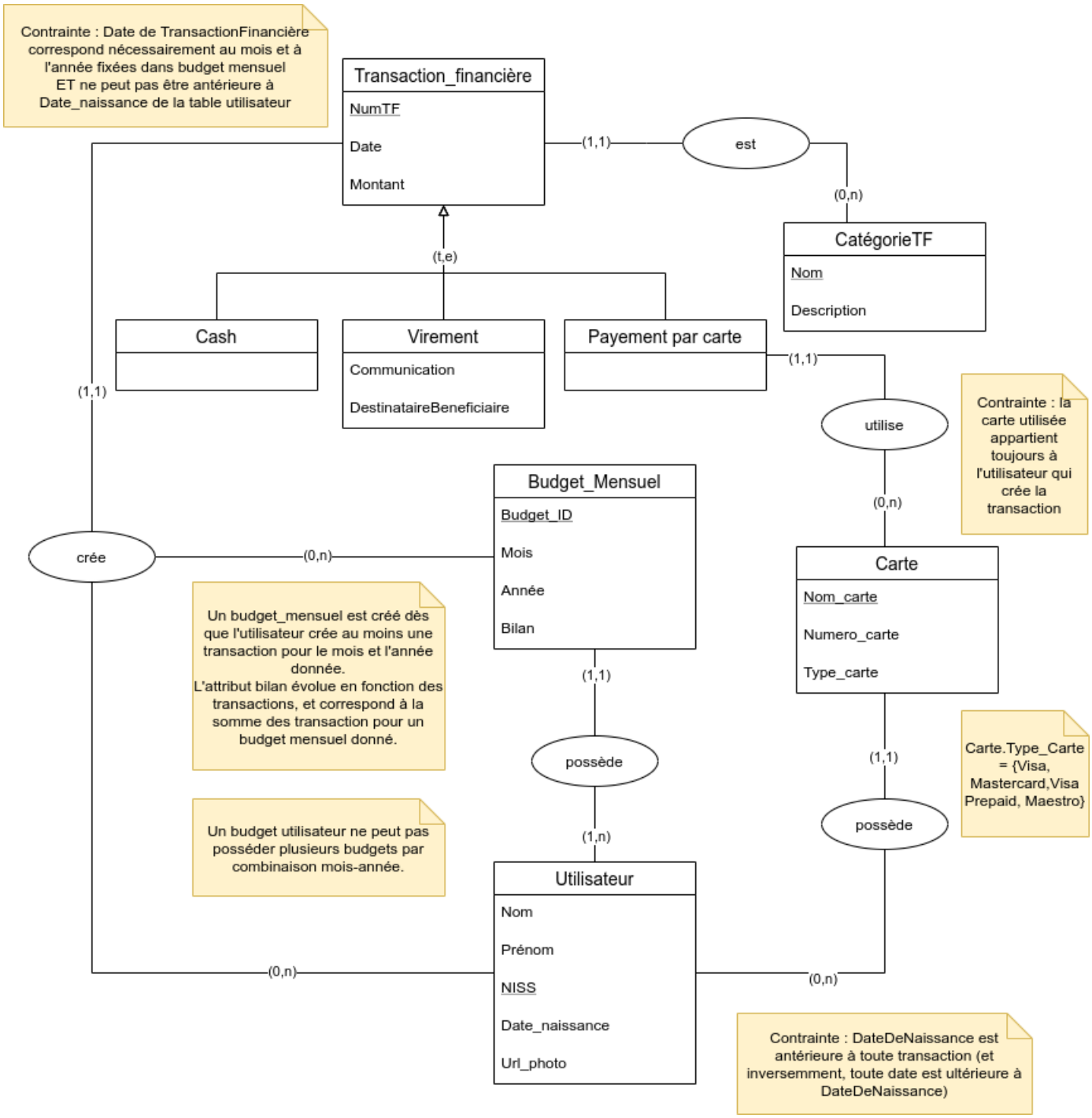

FIGURE 1 – Schéma conceptuel entité-association de la base de données exploitée par Budget Squirrel

3 Schéma logique relationnel

Schéma logique relationnel traduisant le schéma conceptuel, avec une explication du choix des clés de traductions (héritage, associations, contraintes...)

! vérifier si tout correspond avec la DB

! ajouter contraintes

! explication matérialisation

```
utilisateur(nom, prenom, niss, date_naissance, photo, nombre_transactions)

carte(nom_carte, numero_carte, type_carte, niss_util, is_deleted)
carte.niss_util référence utilisateur.niss
carte.type_Carte = {Visa, Mastercard,Visa Prepaid, Maestro}

budget_mensuel(budget_ID, mois, annee, statut, bilan, reste, niss_util)
budget_mensuel.niss_util référence utilisateur.niss

transaction_financiere(num_tf, date_tf, montant, budget_ID, categorie)
transaction_financiere.budget_ID référence budget_mensuel.budget_ID
transaction_financiere.categorie référence categorie.nom_cat

tf_cash(num_tf)
tfcash.num_tf référence transaction_financiere.num_tf

tf_virement(num_tf, communication, destbenef)
tfvirement.num_tf référence transaction_financiere.num_tf

tf_carte(num_tf, nom_carte)
tfcarte.num_tf référence transaction_financiere.num_tf
nom_carte référence carte.nom_carte

categorie_tf(nom_cat, description)
```

4 Code SQL

4.1 Description du code de création de la base de données

Description du code sql de création de la db!! La création de la db doit : créer des tables , avoir des contraintes de clé primaire, unique et externe, avoir des contraintes sous forme de prédicat de table et de colonne, avoir des procédures stockées et des déclencheurs

4.2 Éléments de SQL avancé

4.2.1 Contraintes garanties par des check

4.2.2 Requêtes de consultation

4.2.3 Requêtes de mise à jour

4.2.4 Vues utilisées

Ce qu'on utilise pour l'historique et les différentes statistiques

4.2.5 Déclencheurs

Triggers à utiliser, sans doute, pour le calcul des bilans, et autres

4.2.6 Droits d'accès

Voir si les droits d'accès peuvent être utilisés comme dans le rapport publié par Servais, pour l'instant ça reste un accès root défini par une session > problématique?

5 Description de l'application Web

L'application web que nous avons développé a pour objectif de permettre aux utilisateurs de s'inscrire, d'enregistrer leurs transactions, de consulter leur historique par mois, ainsi que des statistiques sur la répartition de leurs dépenses, de leur revenus, et un bilan total de leur budget. L'accès à la base de donnée est restreint : ce n'est pas un accès root, mais bien un accès spécifiquement défini pour les utilisateurs de l'application. Nous en avions fait la description détaillée à la section précédente.

À l'ouverture de l'application (`localhost/budgetsquirrel/index.html` ou simplement `localhost/budgetsquirrel/`), l'utilisateur a deux possibilités : se connecter, ou s'inscrire.

S'il choisit de s'inscrire, il est redirigé vers un formulaire d'inscription où il doit rentrer son nom, son prénom, son NISS, et choisir une photo de profil. Si un utilisateur est déjà inscrit sous le NISS entré, il en est notifié (et a la possibilité de se rediriger vers la page de connexion). Si le NISS n'est pas encore utilisé par un utilisateur, l'enregistrement se fait, l'utilisateur est notifié du succès de ce dernier, et peut se rediriger vers la page de connexion. La vérification de la présence d'un utilisateur utilisant déjà le NISS se fait à deux niveaux : au niveau de la base de donnée en elle-même, comme nous l'avons vu au dessus, grâce à la contrainte de clé primaire, et également en php, en faisant une sélection de la table utilisateur (si la sélection par rapport au niss entrée renvoie un résultat différent de zéro, l'utilisateur est notifié de l'impossibilité de créer un compte avec ce NISS).

La page de connexion, par souci de simplification, est une simple sélection de profil : l'utilisateur a accès à la liste des utilisateurs, sélectionne son profil (nom et prénom) dans la liste , et se connecte. Ensuite, le NISS de l'utilisateur est récupéré par une méthode POST, enregistré par la méthode SESSION et appelé grâce à `session_start();` à chaque page de l'application où il est enregistré comme connecté).

Une fois qu'il a sélectionné son profil et choisi de se connecter, l'utilisateur est redirigé vers une page d'accueil. La page d'accueil présente brièvement les différentes pages de l'application. Depuis la barre de navigation, présente sur toutes les pages (sauf celles ou l'utilisateur n'est pas considéré comme connecté), l'utilisateur a accès : à la page d'accueil, à l'historique, à la page d'enregistrement, à la page de statistiques, et à sa page personnelle de profil. Il peut ainsi naviguer librement entre les différentes pages.

La page personnelle de profil permet à l'utilisateur de visualiser ses informations : nom, prénom, NISS, date de naissance. Il peut également y ajouter et y supprimer ses cartes [2] . Les listes de ses cartes disponibles et de ses anciennes cartes sont également

2. Cette suppression est une suppression logique du côté de la base de données : les cartes possèdent une colonne is_deleted , par défaut la valeur de la colonne est à 0, et si l'utilisateur « supprime »sa carte, la valeur passe à 1, et la carte est considérée comme virtuellement supprimée

visibles depuis la page de profil. Il peut également y actualiser sa photo.

La page d'enregistrement permet d'enregistrer des transactions. L'utilisateur est obligé d'entrer un montant, de sélectionner une date, une catégorie de transaction, et un type de transaction (à nouveau, la vérification se fait une fois en php, et une fois aussi du côté de la base de données). En fonction du type de transaction effectuée, il doit également ajouter des informations supplémentaires. Le payement en liquide ne demande aucune information supplémentaire, le virement bancaire demande d'introduire un destinataire ou un bénéficiaire, et éventuellement une communication, et un payement par carte nécessite de sélectionner la carte utilisée. Cet aspect est géré uniquement en php, et demande donc de faire deux opérations consécutives si l'on veut rentrer le type de transaction sans passer par l'application Web.

La page historique permet à l'utilisateur de visualiser les transactions effectuées : pour cela, il doit d'abord sélectionner le mois et l'année dont il veut consulter le budget. Il a alors accès à un tableau affichant le montant, la date, la catégorie, et le type de transaction effectuée, et éventuellement la carte utilisée, le destinataire/bénéficiaire, et la communication. En dessous de la table, l'utilisateur peut également voir le bilan total du mois. Enfin, il peut également choisir de supprimer une des transactions affichées.

L'écran de statistiques offre différentes informations sur les données concernant l'utilisateur. Les informations offertes sont les suivantes : un total des dépenses, un total des revenus, et un bilan, mais aussi un tableau montrant la répartition des transactions (dépenses et revenus) par mois, ainsi que le bilan par mois, pour une vue plus condensée de l'historique, un tableau montrant la répartition totale par catégorie (couplé à une description de chaque catégorie), et enfin la répartition des transactions entre les différents types de payements (le nombre d'utilisations, le bilan des dépenses et le bilan des revenus pour chaque type de transaction).

À tout moment, l'utilisateur peut également se déconnecter, en cliquant, en haut à droite de la barre de navigation, sur "déconnexion". Les informations de sessions (donc, le NISS sous lequel l'utilisateur est connecté) sont effacées, et l'utilisateur peut choisir de retourner à l'écran d'accueil.

6 Développement du projet

Quelques mots sur la logique du travail : paired programming, outils employés, défis rencontrés, ...

Le php étant un langage très flexible, il nous a été relativement facile de créer de nombreuses contraintes, et de combiner différentes requêtes SQL pour que l'application réponde exactement comme il faut à tout input de l'utilisateur. Un des défis que nous avons rencontré durant le développement de ce projet a été l'écriture de données en utilisant du SQL pur, sans l'aide de l'application. Certains éléments, facilement mis en place en php, étaient difficiles à traduire en SQL,

7 Conclusion

Les points forts et faiblesses d'application, le temps nécessaire pour développer le concept avec deux personnes, les sources consultées pour la partie pratique (i.e. les slides, ou techniques vues au TP, ainsi que les conseils d'assistante, mais aussi w3schools ou bien stackoverflow). Finalement quelques mots sur l'expérience du projet.