



UNIVERSITÉ LIBRE DE BRUXELLES



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	2
1.1	Domaine d'application	2
2	Schéma conceptuel	3
3	Schéma logique relationnel	4
4	Code SQL	5
4.1	Description du code de création de la base de données	5
4.2	Éléments de SQL avancé	5
4.2.1	Contraintes garanties par des check	5
4.2.2	Requêtes de consultation	5
4.2.3	Requêtes de mise à jour	5
4.2.4	Vues utilisées	5
4.2.5	Déclencheurs	5
4.2.6	Droits d'accès	5
5	Description de l'application Web	6
6	Développement du projet	8
7	Conclusion	9

1 Introduction

Si besoin de citer : (**SOURCE**) Notes de bas de page¹

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

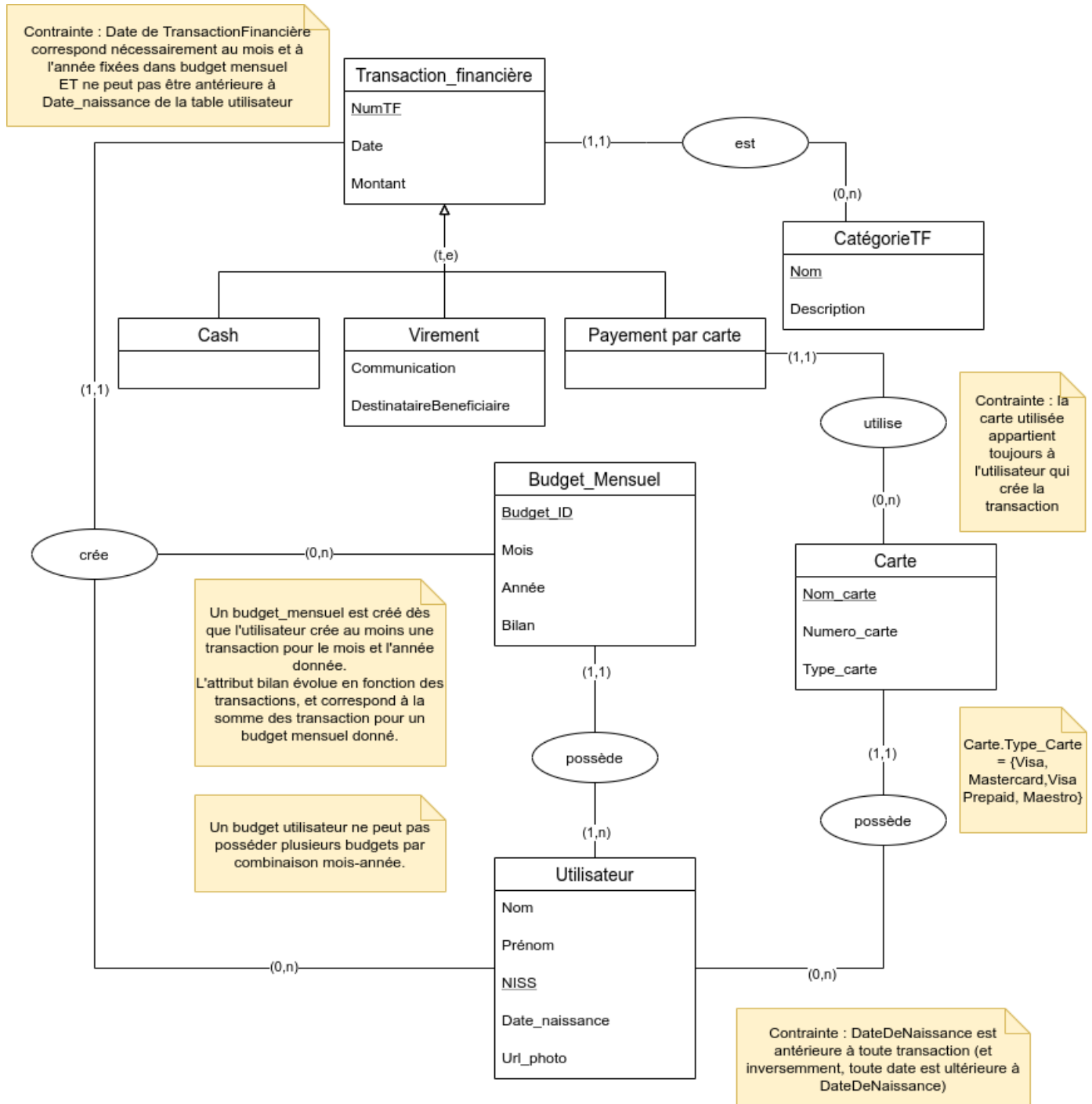


FIGURE 1 – Schéma conceptuel entité-association de la base de données exploitée par Budget Squirrel

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 développée 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 revenu, et un bilan total de leur budget.

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 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 (à ce niveau, le niss de l'utilisateur est récupéré par une méthode POST, enregistré par une méthode SESSION et appelé grâce à `session_start()` ; aux différentes pages de l'application).

NOTE : A AJOUTER DANS LA DB : UN ACCES UTILISATEUR A UTILISER PARTOUT DANS L'APP vs un autre accès admin - DB à utiliser slmt au niveau de la db (qui peut par ex. supprimer des utilisateurs et ajouter de nouvelles catégories) comme ça , on exploite aussi les droits d'accès qui sont demandés par le projet et on évite l'accès root, en délimitant ce que l'utilisateur peut consulter et modifier

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 où l'utilisateur n'est pas 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.

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 cartes². Les listes de ses cartes disponibles et de ses anciennes cartes sont également 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,

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

et un type de transaction. En fonction du type de transaction effectué, il doit également ajouter des informations supplémentaires. Le paiement en liquide ne demande aucune information supplémentaire, le virement bancaire demande d'introduire un destinataire ou un bénéficiaire, et éventuellement une communication, et un paiement par carte nécessite de sélectionner la carte utilisée.

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 paiements (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, ...

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.