Laboratoires d'Analyse et gestion de données:

Application Fichiers en langage C
2018-2019
« QuizUp_Like »

I Description :

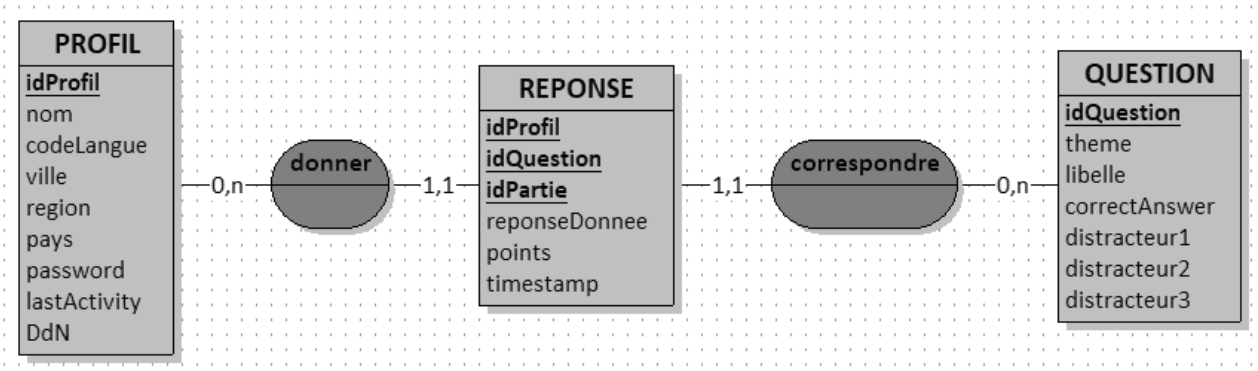
Suite à l'énoncé d'analyse proposé lors des laboratoires de la première partie du cours d'Analyse et Gestion de Données, nous avons conçu un 'gros' modèle conceptuel de données permettant de représenter l'ensemble des données pertinentes gravitant autour du sujet dont un extrait de l'énoncé est rappelé ci-dessous :

> A des fins d'ingénierie inverse dans l'optique d'apporter des améliorations au produit, on désire modéliser l'architecture de données d'une plate-forme de quiz sur smartphone.
> Vous disposez principalement de deux ressources :
> - L'application QuizUp disponible gratuitement sur les stores Apple et GooglePlay
> - Le site QuizUp.com
>
> Votre mission, dans le cadre de cet énoncé, est de réaliser un Modèle Conceptuel de Données permettant de valoriser au mieux l'ensemble des données disponibles dans ces différents documents, en gardant à l'esprit la volonté d'obtenir un modèle modulaire et évolutif.
> Les ressources mises à disposition ne suffisent évidemment pas à la réalisation du travail demandé. Il vous appartiendra de poser toutes les questions nécessaires à votre titulaire de laboratoire, qui agit en tant que représentant/relais avec le client, ainsi que de consulter avidement les ressources pertinentes disponibles sur Internet.
> Etc…

Le modèle conceptuel complet est fourni en Annexe.

I.2. Solution réduite à considérer

[!!! Modèle '~~conceptuel~~' bâtard !!!]



II Définition des structures à utiliser :

```
struct date
{
     short          jour;
     short          mois;
     short          annee;
};
```

Profil

```
struct profil
{
     long           idProfil;
     char           nom[50];
     char           codeLangue[3];
     char           ville[50];
     char           region[50];
     char           pays[50];
     char           password[50];
     time_t         lastActivity;
     date           DdN;
};
```

Reponse

```
struct reponse
{
     long           idProfil;
     long           idQuestion;
     long           idPartie;
     short          reponseDonnee;
     short          points;
     time_t         timestamp;
};
```

Question

```
struct question
{
     long           idQuestion;
     char           theme[50];
     char           libelle[144];
     char           correctAnswer[50];
     char           distracteur1[50];
     char           distracteur2[50];
     char           distracteur3[50];
} ;
```

III Organisation physique des fichiers :

Les données sont mémorisées dans trois fichiers : `profils.dat`, `reponses.dat`, `questions.dat`.

- *Profils* :
  - **Organisation séquentielle logique** (chaînée) :
    Le chaînage permettra de suivre les profils d'utilisateurs triés par ordre alphabétique sur le nom.
- *Réponses :*
  - **Organisation séquentielle physique** (tas).
- *Questions :*
  - **Organisation indexée directe** :
    L'index sera créé au début de l'exécution du programme.
    Cet index contiendra au moins l'identifiant, le thème et le libellé associé à chaque question.
    D'autres informations peuvent être utiles ou nécessaires.

IV Fonctionnalités du programme :

Profils [SCRUD], Reponses [SCR~~UD~~], Questions [SCRUD].

- *Création des fichiers :*
  Bidonnage (préallocation de l'espace disque) du fichier `Questions`.
  On admettra la taille suivante pour le fichier :

Nom du fichier	Nombre maximum d'enregistrements
`questions`	5.000

  La taille des fichiers `profils` et `reponses` variera en fonction des besoins.

- *Profils [SCRUD]* :
  - Ajout d'un nouveau profil utilisateur.
  - Affichage des profils triés par ordre alphabétique croissant sur le nom.
  - Recherche de la fiche d'un utilisateur dont on donne le nom.
  - Modification de la fiche d'un utilisateur dont on donne le nom.
  - Suppression d'un utilisateur.

Attention au respect de l'intégrité référentielle entre `utilisateurs` et `reponses`.

- *Questions [SCRUD] :*
  - Ajout d'une nouvelle question.
  - Affichage de l'ensemble des questions, triées par ordre alphabétique croissant sur le thème et le libellé.
  - Affichage de l'ensemble des thèmes existants dans le fichier.
  - Recherche et affichage des questions associées à un thème donné.
  - Recherche des questions dont le libellé contient un mot donné.
  - Modification de la fiche d'une question dont la fiche a été préalablement recherchée.
  - Suppression de la fiche d'une question dont la fiche a été préalablement recherchée.
  - BONUS : Implémenter une fonctionnalité permettant d'afficher la liste des questions qui n'ont encore jamais été posées (qui ne sont associées à aucune réponse).

Attention au respect de l'intégrité référentielle entre `questions` et `reponses`.

- *Réponses [SCR~~U~~D] :*
  - Ajout de nouvelles réponses :
    - Pour ajouter de nouvelles réponses, l'utilisateur commence par sélectionner son profil ; il indique son nom et son mot de passe, l'application vérifie l'existence et la correspondance de ceux-ci) ;
    - L'application présente ensuite une liste des thèmes disponibles, parmi lesquels l'utilisateur doit choisir ; Le premier choix sera '`Tous les thèmes`' ;
    - Un numéro unique identifiant la partie est généré. L'application génère aléatoirement une série de cinq questions dans le thème choisi. Pour chacune :
      - Le libellé est présenté à l'utilisateur, ainsi que les quatre réponses possibles, affichées dans un ordre aléatoire ;
      - L'utilisateur choisit la réponse qu'il souhaite donner ;

        Il peut également choisir la valeur `-1` pour quitter la partie en cours. Dans ce cas, les questions qui n'ont pas encore reçu de réponse seront enregistrées dans le fichier `reponses` avec des valeurs indiquant une absence de valeur pour `reponseDonnee`, `points` et `timestamp`.
      - L'application retient la réponse donnée dans un nouvel enregistrement du fichier `reponses`.

      Un timestamp `t1` est pris juste avant l'affichage de chaque question, un second timestamp `t2` est pris au moment où l'utilisateur valide son choix. `t1` sera enregistré avec la réponse, la différence `t2-t1` servira à calculer les points récoltés pour une réponse correcte :
      - 10 points pour une réponse dans un intervalle de [0 à 2[secondes,
      - 9 points pour une réponse dans un intervalle de [2 à 3[secondes,
      - 8 points pour une réponse dans un intervalle de [3 à 4[secondes,
      - …
      - 1 point pour une réponse dans un intervalle de [10 à 11[secondes,
      - aucun point au-delà.
  - Affichage des parties jouées par un utilisateur donné (recherche par nom) avec le total des points de chaque partie.
  - Affichage des thèmes auxquels un utilisateur donné (recherche par nom) a participé avec le total des points récoltés par cet utilisateur pour ce thème.
- L'application permettra à l'utilisateur d'utiliser les fonctionnalités décrites en garantissant la cohérence et l'intégrité des données à chaque instant. On permettra également à l'utilisateur de quitter –proprement– l'application.

V Contraintes à respecter :

- *Profils :*
  - Tous les attributs d'un profil sont des champs obligatoires, sauf `ville` et `region`.
  - `nom` est une valeur unique sensible à la casse dans le fichier des profils.
  - `nom` et `password` sont des chaînes de caractères imprimables.
  - A l'affichage d'un profil, la valeur de `password` est remplacée par une suite de `*`.
  - `codeLangue` est une chaîne de caractères qui prend ses valeurs dans la liste suivante : `{FR,EN,DE,NL}`, stockée et affichée en majuscules.
  - `ville` et `region` sont des chaînes de caractères pouvant contenir des lettres minuscules ou majuscules, des chiffres, des espaces et les symboles `_-.'`, stockées et affichées en minuscule, avec uniquement la première initiale en majuscule.
  - `lastActivity` est une valeur continuellement mise à jour avec le timestamp relevé lors de la dernière activité de l'utilisateur (typiquement quand le profil est créé ou modifié, l'utilisateur donne une réponse à une question d'une partie).
  - `lastActivity` est stocké sous la forme d'un `time_t` et est affiché sous la forme d'une chaîne de caractères compréhensible par un humain.
  - `DdN` est une date strictement inférieure à la date du système.
  - ATTENTION, QuizUp_Like est un jeu accessible uniquement aux joueurs âgés de plus de 13 ans.
- *Réponses :*
  - Tous les attributs d'une réponse sont des champs obligatoires, sauf `points` et `timestamp`.
  - Un couple de valeurs `(idProfil,idPartie)` est unique : un utilisateur participe une seule fois à une même partie.
  - Un couple de valeurs `(idQuestion,idPartie)` est unique : une question n'existe qu'une seule fois au cours d'une même partie.
  - `idPartie` est une valeur associée à cinq questions, qui ne se répètera pas par la suite.
  - A l'affichage, `idProfil` est remplacé par le nom du profil dont c'est l'identifiant.
  - `reponseDonnee` est un entier court prenant ses valeurs dans `{-1,1,2,3,4}`. La valeur `-1` est réservée à matérialiser une absence de réponse.
  - `points` est un entier court prenant ses valeurs dans `{-1,0,1,…,9,10}.}`. La valeur `-1` est réservée à matérialiser une absence de réponse.
  - `timestamp` est stocké sous la forme d'un `time_t` dont la valeur est celle du timestamp relevé lorsque l'utilisateur donne la réponse.
- *Questions :*
  - Tous les attributs d'une question sont des champs obligatoires.
  - `theme` est une chaîne de caractères pouvant contenir des lettres minuscules ou majuscules, des chiffres, des espaces et les symboles `_-.'`. Il est stocké et affiché en minuscules.
  - `Libelle`, `correctAnswer`, `distracteur1`, `distracteur2` et `distracteur3` sont des chaînes de caractères imprimables.
  - `distracteur1`, `distracteur2` et `distracteur3` sont des chaînes de caractères différentes.
- Chaque date sera encodée en une seule saisie dans un champ au format JJ/MM/AAAA.
  Les dates devront être valides.
- Des contraintes supplémentaires sont laissées à l'appréciation de l'étudiant.
  Il en sera tenu compte lors de l'évaluation de l'application.
- Les structures fournies dans l'énoncé ne peuvent pas être modifiées.
  D'autres types de données structurés peuvent être définis en fonction des besoins mis en évidence lors de l'implémentation des fonctionnalités demandées.

VI Consignes du travail :

Échéances

- Version électronique :

  Vous enverrez votre travail par courrier électronique à votre titulaire de laboratoire à son adresse HEPL (de la forme `pierre.sagot@hepl.be`) pour le samedi 04 mai 2019, 23h59.

  Votre message aura comme objet '`Groupe NOM Prénom – applic AGD`' et comportera en pièce jointe un fichier ZIP (`GroupeNomPrénom.zip`) reprenant l'ensemble des fichiers et codes de votre application.

  Vous recevrez une confirmation de bonne réception de la part de votre titulaire *dans un délai raisonnable.*

- Version papier :

  Vous remettrez en main propre à votre titulaire de laboratoire, le jour de la présentation de votre application, un dossier écrit qui comportera uniquement :

  - une page de garde avec Groupe, NOM, prénom, année d'étude, intitulé du cours et nom du professeur titulaire,
  - cet énoncé,
  - la définition des structures utilisées dans votre programme, avec pour chaque champ, un commentaire sur leur utilité et les différentes contraintes s'y appliquant.

- Le non-respect des échéances et/ou des modalités de remise entraînera d'office une cote nulle.

Consignes

- Travail individuel. Toute ressemblance avec le travail d'un condisciple donnera lieu à une note de 0.
- Vous fournirez un ensemble de fichiers de test dans lesquels vous aurez inséré les données fournies en annexe.
- La cotation tiendra compte de la présentation (en ce compris l'orthographe) et de la convivialité.
- La mise en pratique des organisations de fichiers imposées ainsi que l'exécution correcte de toutes les fonctionnalités décrites dans l'énoncé font partie des critères d'évaluation.
- Ce dossier fait l'objet d'une évaluation orale, tout étudiant qui remet le dossier mais est absent pour la défense orale recevra une cote nulle.
- Vous êtes susceptibles d'être évalués sur une machine de laboratoire. L'environnement de développement utilisé sera donc en tout point semblable à celui disponible aux laboratoires. Un travail non compilable et/ou non présentable sur une machine de laboratoire recevra une cote nulle.