# INFO-H-303 : Bases de données
# Projet : Agrégateur/créateur de flux d'informations

Professeur : Esteban Zimányi
Assistants : Frédéric Servais et Boris Verhaegen
http://cs.ulb.ac.be/public/teaching/infoh303

Année académique 2009-2010

---

## Étude de cas

On vous demande de créer une application de consultation et de création de flux d'informations RSS 2.0. Cette application est inspirée par *Google Reader*[1].

RSS 2.0[2] (*Really Simple Syndication*) est un format XML utilisé pour la syndication de contenu Web. L'usage le plus courant est de recevoir la liste des articles récents de sites d'informations comme des blogs afin de gagner du temps (ou en perdre moins).

Un utilisateur de l'application sera identifié par son adresse email. La base de données retiendra son surnom, la ville et le pays de son domicile, son avatar (image ou photo), une petite biographie optionnelle, la date à laquelle il s'est inscrit au système ainsi qu'un mot de passe permettant de l'authentifier.

Chaque utilisateur pourra souscrire à des flux d'informations au format RSS 2.0. Un flux possède un nom (ou titre), une brève description, un lien vers le site web associé (par exemple, `http://www.macbidouille.com`) et sera identifié par une url (par exemple, `http://feeds.macbidouille.com/macbidouille/`). D'autres attributs optionnels existent dans la spécification[3], à vous de décider s'ils sont utiles dans votre application. Ces flux sont composés de publications qui possèdent un titre, un lien, une date (timestamp) de publication et une description plus ou moins brève. Pour chacune des publications des flux auxquels un utilisateur a souscrit, le système retiendra si l'utilisateur a lu cette publication afin de pouvoir ne lui présenter que celle qu'il n'a pas encore lues. Lors de la première inscription à un flux, le système sauvegardera la liste des 10 dernières publications de ce flux. Ensuite, à chaque consultation d'un flux par un utilisateur, le système téléchargera et stockera les nouvelles publications.

Lorsque l'utilisateur consulte les publications de ses flux, il lui est possible de partager celles de son choix et peut adjoindre à ces dernières un court commentaire. L'ensemble des publications partagées par un utilisateur forme un nouveau flux, le flux de l'utilisateur. Tout utilisateur peut souscrire à ce nouveau flux comme pour tout autre flux.

Chaque utilisateur a également un ensemble d'amis. Les relations d'amitiés sont bidirectionnelles : si Alice est l'amie de Bob, alors Bob est l'ami d'Alice. Pour ajouter un ami, l'utilisateur rentre l'adresse email de celui-ci, l'ami reçoit une notification de cette demande qu'il peut accepter ou refuser. Chaque utilisateur est automatiquement inscrit aux flux de ses amis. Enfin le système donne la possibilité aux utilisateurs de commenter les publications partagées par leurs amis. Un seul commentaire par publication et par utilisateur est permis. Ces commentaires sont visibles (avec la date de leur création) uniquement par les amis de la (les) personne(s) ayant partagé la publication concernée.

[1] `http://www.google.com/reader/`
[2] `http://en.wikipedia.org/wiki/RSS`
[3] `http://validator.w3.org/feed/docs/rss2.html`

## Format des données

Vous recevrez plusieurs fichiers RSS 2.0 à importer dans votre application. Ces fichiers respecteront la norme RSS 2.0[4]. Vous n'êtes pas obligés de supporter les éléments optionnels non repris dans l'énoncé du projet. Toutes les dates respectent le format RFC 822[5].

Un fichier XML suivant le modèle ci-dessous, concernant les données de l'application (lectures, amis, partages, etc.) vous sera également fourni.

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
    <user>
        <email>Adresse de l'utilisateur</email>
        <password>Mot de passe (en clair)</password>
        <nickname>Pseudonyme</nickname>
        <city>Ville du domicile</city>
        <country>Pays du domicile</country>
        <avatar>Nom d'un fichier image</avatar>
        <biography>Courte Biographie (optionnelle)</biography>
        <subscribeDate>Date d'inscription<subscribeDate>
    </user>
    ...
</users>
<friends>
    <friend>
        <email>Adresse de l'utilisateur</email>
        <email>Adresse de l'utilisateur</email>
        <date>Date de la demande d'amitié</date>
        <accepted>True ou False</accepted>
    </friend>
    ...
</friends>
<reads>
    <read>
        <email>Adresse de l'utilisateur</email>
        <date>Date de lecture</date>
        <feed>Lien du flux</feed>
        <item>Lien de la publication</item>
    </read>
    ...
</reads>
<comments>
    <comment>
        <email>Adresse de l'utilisateur</email>
        <text>Texte du commentaire</text>
        <feed>Lien du flux</feed>
        <item>Lien de la publication</item>
        <date>Date du commentaire</date>
    </comment>
    ...
</comments>
<shares>
    <share>
        <feed>Lien du flux</feed>
        <item>Lien de la publication</item>
        <date>Date du partage</date>
        <email>Adresse de l'utilisateur</email>
        <text>Texte du commentaire (optionnel)</text>
    </share>
    ...
</shares>
<subscriptions>
    <subscription>
        <email>Adresse de l'utilisateur</email>
        <feed>Lien du flux</feed>
        <date>Date d'abonnement</date>
    </subsscription>
    ...
</subscriptions>
```

On vous demande créer un script d'insertion de ces données dans votre application, à l'aide par exemple de la librairie XML Parser[6].

[4] http://validator.w3.org/feed/docs/rss2.html
[5] http://asg.web.cmu.edu/rfc/rfc822.html
[6] http://php.net/manual/en/book.xml.php

# Déroulement du projet

## Première partie

Pour cette partie, on vous demande de modéliser le problème à l'aide du formalisme entité-association et de préciser les contraintes d'intégrité nécessaires. Ces contraintes doivent être exprimées en français et utiliser les mêmes noms d'entités, d'associations ou d'attributs que dans votre modèle conceptuel. Vous pouvez également exprimer et justifier des hypothèses sur votre modèle. Ces hypothèses peuvent résulter par exemple d'ambiguïtés dans l'énoncé du problème.

Déduisez ensuite de ce modèle conceptuel le modèle relationnel correspondant ainsi que ses contraintes. Vos choix de modélisation doivent être justifiés.

Ces modèles doivent être suffisamment riches pour pouvoir servir de support à l'application décrite ci-dessous ainsi que pour pouvoir répondre aux requêtes de la deuxième partie.

## Deuxième partie

### Création

On vous demande tout d'abord de déduire de votre modèle relationnel un script SQL DDL de création de la base de données et de ses différentes tables ainsi que de créer cette base de données.

### Initialisation

On vous demande d'écrire un script permettant d'importer dans votre base de données les fichiers XML fournis. Ces données devront être présentes dans votre base de données lors de la défense. Les fichiers XML seront disponible le 3 novembre sur la page web du projet.

### Application

Nous vous demandons ensuite de développer une interface graphique pour votre base de données permettant au minimum de réaliser les opérations ci-dessous :

- connecter un utilisateur
- souscrire à un flux RSS 2.0
- visualiser le texte des publications non lues des flux parmi ses souscriptions, en ce compris les flux de ses amis
- lire une publication
- partager une publication en laissant ou non un commentaire
- visualiser le résultat des requêtes ci-dessus à l'endroit où cela vous semble cohérent

Votre application devra bien entendu veiller à ce que la base de données reste cohérente.

Vous pouvez bien sûr ajouter des fonctionnalités à votre application comme par exemple

- afficher des statistiques de manière graphiques (comme *Trends* dans *Google Reader*)
- afficher les images (avatars, images d'une publication, etc.)
- commenter une publication partagée par un ami
- rechercher un nouvel ami, envoyer une demande, accepter une demande
- ajouter un utilisateur
- gérer l'internationalisation
- etc.

Ces apports personnels seront valorisés à hauteur de 4 points sur 20.

### Requêtes

Nous vous demandons d'écrire en algèbre relationnelle et calcul relationnel tuple les requêtes R1, R2, R3 et R4 ainsi que toutes les requêtes en SQL. Si une requête vous semble imprécise, indiquez dans votre rapport les hypothèses que vous avez faites pour lever ces imprécisions.

- R1 : Les bleus : Tous les utilisateurs qui ont au plus 2 amis.
- R2 : Suggestion de flux : La liste des flux auxquels a souscrit au moins un utilisateur qui a souscrit à au moins deux flux auxquels $X$ a souscrit.
- R3 : Nettoyage de flux : La liste des flux auxquels $X$ a souscrit, auxquels aucun de ses amis n'a souscrit et duquel il n'a partagé aucune publication.
- R4 : Suggestion d'ami : La liste des utilisateurs qui ont partagé au moins 3 publications que $X$ a partagé.
- R5 : Statistiques des souscriptions : La liste des flux auquel un utilisateur est inscrit avec le nombre de publications lues, le nombre de publications partagées, le pourcentage de ces dernières par rapport aux premières, cela pour les 30 derniers jours et ordonnée par le nombre de publications partagées.
- R6 : Statistiques des amis : La liste des amis d'un utilisateur avec pour chacun le nombre de publications lues par jour et le nombre d'amis, ordonnée par la moyenne des lectures par jour depuis la date d'inscription cet ami.

## Rapports

Pour la première partie, on vous demande les documents suivants :

- un diagramme entité-association modélisant le projet ainsi que ses contraintes
- une traduction relationnelle de ce diagramme et ses contraintes
- vos hypothèses et la justification de vos choix de modélisation

Le formalisme utilisé doit être un de ceux vus au cours ou aux TPs.

Pour la deuxième partie, vous rendrez un rapport contenant :

- les documents de la première partie tenant compte des remarques des assistants
- le script SQL DDL de création de la base de données
- les requêtes demandées
- les instructions d'installation de votre application
- un scénario de démonstration de votre application
- les explications et justifications de vos choix et hypothèses

Le code source de votre script d'insertion et de votre application devra être rendu par mail à `ulb.code+infoh303@gmail.com` avant votre défense.

## Informations pratiques

- Le projet se fera obligatoirement par groupe de deux.
- La première partie, en version papier, devra être rendue aux assistants pour le mardi 27 octobre à 14h au TP et être présentée la semaine suivante suivant un horaire à déterminer.
- Le rapport de la seconde partie devra être rendu en version papier lors de votre défense. Une archive contenant tous les codes sources devra être envoyée à l'adresse `ulb.code+infoh303@gmail.com` pour le 8 décembre à minuit. La défense du projet aura lieu la même semaine suivant un horaire à déterminer. On vous demandera de présenter votre application pendant 10 minutes et de répondre à quelques questions.
- Le projet comptera pour 25% de la note finale du cours.
- Les apports personnels au projet seront valorisés à hauteur de 4 points sur 20. En d'autres mots, un projet parfait sans apport personnel aura une valeur de 16 points sur 20.
- Sauf mention explicite, vous pouvez utiliser les langages et outils de votre choix (MySQL, PostgreSQL, PHP, Java, Python, ...).
- Vous pouvez développer sur votre propre machine et présenter vos projets sur un ordinateur portable que vous apporterez lors de la défense. Au besoin, nous mettrons à votre disposition une base de données MySQL ainsi qu'un espace web PHP sur un serveur de l'Université.

Bon travail !