TP4

LE LANGAGE SQL (ENQUÊTE SUR LES PANAMA PAPERS).

Table des matières

ECG 2 Maths appliquées, http://louismerlin.fr.

Ce TP fait suite à la partie du cours de présentation des bases de données, sous forme théorique. Nous allons maintenant découvrir le langage SQL qui est construit pour dialoguer avec des bases de données relationnelles.

Nous n'entrerons pas (trop) dans les stratégies de création de données. Nous supposerons que la base de données est déjà construite et déjà remplie. Ce qui nous intéresse ici c'est d'interroger ces données.

Avant de commencer, il faudra télécharger un interpréteur de commandes SQL et la base de données que nous allons utiliser.

1. Le programme à télécharger s'appelle SQLite, il est léger, libre et gratuit, et se trouve ici.
2. La base de données est disponible sur ma page Internet à cette adresse. Elle est écrite dans un format .sqlite3 (qui est en fait un groupe de fichiers .csv), qui est bien sûr compatible avec le logiciel SQLite. Attention, le fichier est gros (environ 80MB).

1. Le contexte

En avril 2016, le journal allemand Süddeutsche Zeitung ainsi que le Consortium International des Journalistes d'Investigation (ICIJ) publient des documents confidentiels provenant d'un cabinet d'avocats panaméen.

Cette publication fait grand bruit à travers le monde, car les documents sur lesquels ont enquêté les journalistes du consortium international révèlent des informations sur plus de 214 000 sociétés offshores ainsi que le nom des actionnaires de celles-ci. C'est l'affaire des Panama Papers.

Si l'affaire a fait tant de bruit, c'est parce qu'elle dévoile un système complexe, massif et secret permettant à des entreprises ou à des particuliers de cacher de grosses sommes d'argent sur des comptes bancaires. Ces comptes sont généralement situés dans des pays où la législation est avantageuse, que ce soit en termes de secret bancaire, de taxation, ou de contrôle de la provenance de l'argent. Ce phénomène est appelé l'évasion fiscale.

Si ces pratiques sont souvent légales, l'opinion publique les voit en général d'un mauvais œil. En effet, des sommes d'argents générées au sein d'un État donné (les bénéfices d'une entreprise par exemple) sont taxées par ce même Etat. Le fruit de cette taxation est redistribué (entre autres) aux services publics dont bénéficie la population du pays en question.

Cependant, l'évasion fiscale consiste à transférer les bénéfices du pays d'origine vers des pays à législation avantageuse (appelés les paradis fiscaux). Ainsi, les sommes d'argent générées dans un pays donné échappent en partie à l'impôt, et ne bénéficient donc plus aux populations locales. Dans certains pays, le montant estimé de l'évasion fiscale est égal ou supérieur au budget annuel de l'Etat, lorsqu'en parallèle leurs hôpitaux peinent à assurer les soins nécessaires (source : ICIJ).

Dans les Panama Papers se trouvaient par exemple les noms de plusieurs responsables politiques à travers le monde. Certains d'entre eux ont dû démissionner suite à la pression de l'opinion publique. Mais les Panama Papers ont aussi mis en lumière des moyens de financement de réseaux criminels ou terroristes.

En France, l'affaire a été révélée par 2 groupes de journalistes : Premières Lignes Production (qui a réalisé ce documentaire), et Le Monde.

2. La base de données.

Les Panama Papers sont composés de près de 11,5 millions de documents (emails, courriers, contrats, etc.), pour un volume d'environ 2 Go. De ces documents écrits, l'ICIJ a tenté d'extraire les informations essentielles grâce à des algorithmes. Le résultat de cette extraction a été placé dans une base de données rendue publique.

Cette base n'est pas exacte, elle contient par exemple beaucoup de doublons et de champs erronés.

Grossièrement, la BDD des Panama Papers contient des sociétés offshores. Celles-ci sont créées pour des bénéficiaires par des fournisseurs de services offshores. Des intermédiaires se chargent généralement de faire le lien entre les bénéficiaires et les fournisseurs de services offshores.

Il y a 4 tables principales dans la base de données.

1. La table **entity**. C'est elle qui contient les sociétés offshores.

2. La table **intermediary**, qui contient les intermédiaires.
3. La table **address** qui contient les adresses de certaines sociétés intermédiaires.
4. La table **officer**, contenant entre autres les bénéficiaires des sociétés.

Ces tables contiennent les données publiées par l'ICIJ, auxquelles ont été ajoutées quelques données fictives spécialement pour ce TP, notamment la société *Big Data Crunchers Limited*. Elle a été créée de toutes pièces pour servir de fil rouge.

Remarque 1. Une société peut être domiciliée dans un pays, mais être enregistrée dans un autre. Dans ce cas, cette société répondra à la juridiction dans laquelle elle est enregistrée, même si son adresse officielle n'est pas dans cette juridiction.

Souvent, les termes *juridiction* et *pays* sont confondus. En général, les lois sont les mêmes à l'intérieur d'un même pays. Mais parfois, un pays possède plusieurs juridictions : c'est souvent le cas des états fédéraux, dans lesquels chaque état possède des lois différentes. Par exemple, l'état du Delaware aux USA est souvent considéré comme un paradis fiscal, car les lois y sont plus avantageuses pour les sociétés que dans les autres états des USA.

3. Objectif du TP

Je vous propose de vous mettre dans la peau d'un enquêteur qui enquête sur le financement d'un réseau criminel.

Vous avez au cours de votre enquête intercepté une facture émise par une mystérieuse société qui s'appelle *Big Data Crunchers Limited*. Sur cette facture, l'adresse de cette société n'est pas indiquée. Vous ne savez pas qui se cache derrière cette société, mais vous pensez qu'elle peut être une société écran. Une société écran ne se crée pas si facilement que cela. En général, il faut demander de l'aide à des services spécialisés. On les appellera ici des intermédiaires.

Vous allez donc enquêter sur cette mystérieuse société, mais aussi sur les intermédiaires qui ont aidé à la créer, car vous pensez qu'il sera peut-être possible de les accuser de complicité.

4. Un peu de vocabulaire

Définition : Société
> En économie, une société est la forme juridique la plus répandue des entreprises ; c'est un terme souvent utilisé pour désigner une entreprise.

Définition : Société offshore
> C'est une société "extraterritoriale" en français. En pratique, il s'agit d'une société créée dans un pays dans lequel le bénéficiaire économique final n'est pas résident et qui est dirigée hors du pays dans lequel elle est immatriculée. Elles sont souvent utilisées dans des pays où la fiscalité est avantageuse. La société offshore est une forme de société écran, qui présente toutes les caractéristiques d'une société réelle (elle est immatriculée par exemple), mais dont l'apparence ne correspond pas à la réalité.

Définition : Société écran
> Une société écran est une société fictive, créée pour dissimuler les transactions financières d'une ou de plusieurs autres sociétés.

Définition : Intermédiaire
> Un intermédiaire est dans la plupart des cas une personne ou un cabinet d'avocats agissant pour des clients recherchant un fournisseur de services offshores ou demandant la création d'une société offshore.

Définition : Fournisseur de services offshore

(en anglais : offshore service provider ou agent) C'est une société qui fournit des services dans une juridiction offshore, sur demande d'un client. Ces services peuvent être la création, l'enregistrement ou la gestion de sociétés offshores.

Définition : Bénéficiaire

en anglais : beneficial owner ou beneficiary) C'est la personne réellement propriétaire de la société. Dans le monde offshore, l'identité du bénéficiaire est souvent gardée secret.

Remarque 2. L'ICIJ tient à préciser à toute personne souhaitant utiliser la base de données les points suivants :

1. L'utilisation de sociétés offshores et de trusts n'est pas toujours illégale. Les personnes, sociétés ou autres entités citées dans la base de données n'ont donc pas forcément enfreint la loi ou agi de manière illégitime.
2. Beaucoup de personnes ou entités ont des noms similaires. Avant de conclure que deux noms correspondent à la même personne ou entité, il est conseillé de vérifier leurs adresses respectives ou toute autre information pertinente.
3. En cas d'erreur dans la base de données, prendre contact avec l'ICIJ.

5. Déroulement du TP

1. Commençons tout d'abord par créer la table **entity** qui accueillera les sociétés offshore contenues dans les Panama Papers.

```
CREATE TABLE entity (
  id INTEGER,
  name TEXT NOT NULL,
  jurisdiction TEXT,
  jurisdiction_description TEXT,
  company_type TEXT,
  id_address INTEGER
  incorporation_date DATE,
  inactivation_date DATE,
  status TEXT,
  service_provider TEXT,
  country_codes TEXT,
  countries TEXT,
  source TEXT,
  PRIMARY KEY(id),
  FOREIGN KEY(id_address) REFERENCES address(id)
)
```

La commande CREATE TABLE permet de créer une table et de renseigner son nom (sur la première ligne) et les différents attributs. Nous devons spécifier le type de chaque attribut, ici du texte (chaîne de caractères), un nombre entier ou une date. D'autres options sont possibles, comme un nombre réel ou un booléen (un objet qui prend 2 valeurs, VRAI ou FAUX).

Le mot-clé NOT NULL nous empêche de créer une ligne sans renseigner le nom de la société.

Enfin, nous avons indiqué la clé primaires de la table et une clé étrangère en l'attribut id-address qui fait référence à la colonne id de la table **address**.

2. Maintenant que nous avons créé la structure de la table, insérons une ligne

```
INSERT INTO entity(id, name, jurisdiction,
    jurisdiction_description, incorporation_date) VALUES
    (0,'Une societe', 'IMG', 'Le pays imaginaire', '
    2023-02-03');
```

Nous spécifions la table à remplir grâce à INSERT INTO, puis nous indiquons entre parenthèse les colonnes que nous voulons compléter, puis nous donnons les valeurs à insérer après le mot clé VALUES. Ces valeurs doivent être dans le même ordre que le nom des attributs.

Si certaines valeurs sont laissées vides, elles ne contiennent aucune valeur.

3. Chercher à quoi sert la commande DELETE FROM et supprimer de la table la ligne que nous venons de créer.

*Nous supposons dorénavant que la table **entity** a été remplie : c'est celle que vous avez téléchargé dans la base de données.*

4. La commande SELECT permet de communiquer avec la base de données. À chaque requête avec SELECT, le SGBDD nous renvoie une table. Exécuter la commande

```
SELECT * FROM entity ;
```

Le caractère * derrière SELECT signifie que nous voulons obtenir toutes les lignes et toutes les colonnes disponibles.

Comparer la commande précédente avec

```
SELECT DISTINCT * FROM entity ;
```

À quoi sert le mot-clé DISTINCT ?

5. Exécuter la commande

```
SELECT id, name, status FROM entity ;
```

Que voyez-vous ? Quelle est la méthode SQL pour obtenir une projection ?

6. Commençons maintenant notre enquête ! Nous allons chercher cette mystérieuse société dont le nom est *Big Data Crunchers Limited*. Il s'agit donc de fabriquer une restriction de la relation **entity**. C'est l'affaire du mot clé WHERE. Exécuter

```
SELECT * FROM entity WHERE name = 'Big Data Crunchers Ltd.';
```

Qu'apprend-on sur la société qu'on cherche ?

Remarque 3. Pour trouver la société *Big Data Crunchers Limited*, nous avons utilisé l'opérateur de comparaison =. D'autres opérateurs de comparaison existent :

- $A = B$: A est égal à B.
- $A <> B$: A est différent de B.
- $A > B$ et $A < B$: A est supérieur/inférieur à B.
- $A >= B$ et $A =< B$: A est supérieur/inférieur ou égal à B.
- A BETWEEN A AND C : A est compris entre B et C.
- A LIKE 'chaine de caractère' : pour comparer A à une chaine de caractère donnée.
- A IN $(B_1, B_2, \cdots)$: A est présent dans la liste $(B_1, B_2, \cdots)$.
- A IS NULL : A n'a pas de valeur.

7. Les opérateurs logiques OR, AND et NOT signifient respectivement OU, ET et NON. Grâce à ces opérateurs, on peut complexifier un peu nos conditions.

Exécuter

```
SELECT * FROM entity
WHERE (id < 10000004 AND (NOT id < 10000000)) OR (name = 'Big Data
    Crunchers Ltd.');
```

et interpréter.

8. Le produit cartésien s'obtient facilement grâce à SELECT :

```
SELECT * FROM entity, address ;
```

Attention le temps de calcul est parfois long.

9. Nous voulons maintenant savoir si *Big Data Crunchers Limited* a servi d'intermédiaire. Les intermédiaires peuvent être soit des personnes physiques, soit des sociétés. Il y a donc peut-être des sociétés qui sont à la fois dans la table **intermediary** et dans **entity**.

Pour utiliser un opérateur binaire (intersection, union, différence) il faut que les tables aient le même schéma, ce qui n'est pas le cas ici.

a. On suppose que deux sociétés qui ont même nom et même adresse sont les mêmes. Faire une projection des tables **intermediary** et dans **entity** pour ne conserver que les attributs name et id_address.

b. Pour avoir la liste des sociétés de **entity** et des intermédiaires, on utilise le mot clé UNION. Exécuter

```
SELECT name, id_address FROM entity
UNION
SELECT name, id_address FROM intermediary ;
```

c. Utiliser le mot clé EXCEPT pour trouver les sociétés qui ne sont pas des intermédiaires.

d. Utiliser enfin le mot clé INTERSECT pour trouver les sociétés qui sont aussi des intermédiaires. Chercher si *Big Data Crunchers Limited* en fait partie.

e. (∗) Imaginons que deux tables présentent un grand nombre d'attributs et on veut savoir si une ligne d'une table se trouve aussi dans la deuxième table, sans avoir à comparer tous les attributs un par un.

Trouver une commande SQL qui réponde à ce problème.

10. Nous voulons maintenant trouver l'adresse de la mystérieuse société *Big Data Crunchers Limited*.

a. Nous allons déjà faire la jointure de la table **entity** avec la table **address** Pour cela, il faut exécuter la commande

```
SELECT *
FROM entity
  JOIN address ON entity.id_address = address.id_address
    ;
```

Les tables à joindre sont en effet **entity** et **address** (dans cette ordre) et la condition de jointure apparaît après le mot-clé ON. Si la condition de jointure porte sur un groupe d'attributs plus grand, on utilise la syntaxe

```
SELECT * FROM t1 JOIN t2 ON (t1.fk1 = t2.pk1 AND t1.fk2
    = t2.pk2);
```

b. À l'aide de l'exercice du cours, expliquer pourquoi la commande

```
SELECT * FROM entity, address WHERE entity.id_address = address.
    id_address ;
```

a le même effet que la commande de jointure.

c. Retrouver maintenant l'adresse de la société mystère.

11. Retrouvons maintenant les intermédiaires qui ont participé à la création de la société *Big Data Crunchers Limited*.

a. Quel est le rôle de la table **assoc_inter_entity** ?

b. Exécuter et interpréter la commande

```
SELECT
    i.id as intermediary_id,
    i.name as intermediary_name,
    e.id as entity_id,
    e.name as entity_name,
    e.status as entity_status
FROM
    intermediary i,
    assoc_inter_entity a,
    entity e
WHERE
    a.entity = e.id
    AND a.inter = i.id
    AND e.name = 'Big Data Crunchers Ltd.' ;
```

Le mot-clé as permet de renommer les attributs en quelque chose de plus lisible. De même, les lettres a, e et i dans le FROM sont aussi des alias.

c. Conclure sur les intermédiaires qui ont servi à la création de *Big Data Crunchers Limited*.

12. Nous voulons maintenant incriminer les sociétés qui ont bénéficié des services des deux intermédiaires que nous avons trouvés, dans chacune des juridictions. Nous devons donc faire des agrégations.

a. Exécuter la commande

```
SELECT status, count(*) FROM entity GROUP BY status ;
```

Ici nous avons placé l'attribut de partitionnement status derrière le mot-clé GROUP BY et la fonction d'agrégation count() dans le SELECT. Que renvoie cette commande ?

b. Exécuter et interpréter

```
SELECT max(incorporation_date) AS maxi FROM entity;
```

La fonction max est une autre fonction d'agrégation.

c. Comparer les requêtes

```
SELECT status, count(*) FROM entity GROUP BY status ;
```

et

```
SELECT count(*) FROM entity GROUP BY status ;
```

Quel est l'intérêt de la première option ?

d. Exécuter et interpréter la commande

```
SELECT
    i.id as intermediary_id,
    i.name as intermediary_name,
    e.jurisdiction,
    count(*)
FROM
    intermediary i,
    assoc_inter_entity a,
    entity e
WHERE
    a.entity = e.id
    AND a.inter = i.id
    AND (i.id = 5000 OR i.id = 5001)
GROUP BY
    i.id, i.name, e.jurisdiction;
```

13. Enfin, il est parfois utile de savoir réorganiser les lignes d'une table selon le critère que l'on choisit. On utilise pour ça le mot clé ORDER BY comme dans la commande

```
SELECT * FROM entity ORDER BY lifetime ;
```

a. Que constatez-vous lorsqu'on exécute

```
SELECT * FROM entity ORDER BY lifetime DESC;
```

b. Enfin, exécuter la commande

```
SELECT
    i.id AS intermediary_id,
    i.name AS intermediary_name,
    e.jurisdiction,
    e.jurisdiction_description,
    count(*) as cnt
FROM
    intermediary i,
    assoc_inter_entity a,
    entity e
WHERE
    a.entity = e.id AND
    a.inter = i.id AND
    (i.id = 5000 OR i.id = 5001)
GROUP BY
    i.id, i.name, e.jurisdiction, e.
    jurisdiction_description
ORDER BY
    cnt DESC ;
```

Qu'en déduisez-vous ?

6. Conclusion : les commandes à retenir

- La commande CREATE TABLE.
- Les mots clés PRIMARY KEY et FOREIGN KEY avec la méthode pour pointer vers une clé candidate d'une autre table.
- Les stratégies de projection restriction et produit cartésien avec SELECT.
- Pour la restriction, les opérateurs de comparaison.
- Les opérateurs logiques OR, AND et NOT.
- Le mot clé DISTINCT.
- Les opérations binaires sur les tables données par les mots clé UNION, INTERSECT, EXCEPT.
- La méthode de jointure.
- Le mot-clé GROUP BY pour faire un agrégat.
- Les fonctions d'agrégation count(*), min, max, avg, sum (compter, minimum, maximum, moyenne et somme).
- Le mot-clé ORDER BY.