Bases de données

Un aperçu de SQL par l'exemple

SQL_00
v130b

2022-01-16

UNIVERSITÉ DE SHERBROOKE

Christina.Khnaisser@USherbrooke.ca
Luc.Lavoie@USherbrooke.ca

Plan

- Premiers repères SQL
  - Transposition des concepts relationnels
  - Structure du langage
- Exemple : l'évaluation à l'UdeS
  - Rappels
    - Modèle relationnel
    - Exemple de données
  - Développement itératif
    - Étapes 0, 1, 2, 3
  - Essai
    - Insertion de données
    - Requêtes
- Les colles du prof

Premiers repères SQL

- Correspondance avec le modèle de Date
- Niveaux schématiques

Modèle et langage SQL
Correspondance avec le modèle de Date

Relvar	**TABLE**	Un prédicat modélisé par un ensemble de tuples représenté par une table.
Tuple	**ROW**	Une proposition modélisée par un ensemble d'attributs représentée par une ligne.
Attribut	**COLUMN**	Une caractéristique typée et valuée désignée par un nom représentée par un champ.
Type de base	**TYPE**	Un ensemble des valeurs propres.
Sous-type	**DOMAIN**	Un sous-ensemble d'un type défini par un domaine.
Degré	...	Le nombre de champs d'une ligne ou de colonnes d'une table... disponible via le catalogue maintenu par le SGBD.
Cardinalité	**COUNT(*)**	Le nombre de lignes d'une table peut être obtenu par la fonction de dénombrement.
Modèle logique	...	Un ensemble de définitions chacune uniquement identifiée... c'est le catalogue de la base de données, maintenu par le SGBD.
Base de données	**DATABASE**	Un ensemble de variables correspondant à un modèle logique.

clé (par rapport à une table)

sous-ensemble d'attributs déterminant un tuple unique au sein de la table.

- clé candidate
  - clé irréductible.
- clé primaire
  - clé candidate singularisée parmi les autres clés candidates.
- clé secondaire (ou unique)
  - toute clé candidate qui n'est pas la clé primaire.

***Note** : Les notions relationnelles de superclé et de clé candidate sont souvent confondues dans les présentations de SQL ; ce ne sera pas le seul écart par rapport à la théorie relationnelle, malheureusement!*

- clé référentielle
  - sous-ensemble d'attributs d'une table (la référente)
  - qui est une clé candidate dans une autre table (la référée)
  - et pour laquelle la contrainte suivante est respectée : toute valeur de la clé référentielle présente dans la table référente l'est aussi dans la table référée.

Modèle et langage SQL
Niveaux schématiques

Logique

- **LDD (définition)**
  - **définition du modèle logique**
  - **tables**
  - **types**
  - **routines**
- **LMD (manipulation)**
  - **mise-à-jour des données**
  - **évaluation des données**
- *LCD (contrôle)*
  - *contrôle d'accès aux données*
  - *contrôle des transactions*

- **DDL (definition)**
  - **Create, Alter, Drop**
  - **Table, View…**
  - **Domain, Type…**
  - **Function, Procedure, Trigger**
- **MDL (modification)**
  - **Insert, Update, Delete**
  - **Select**
- *CDL (control)*
  - *Grant, Revoke…*
  - *Begin, Commit, Rollback…*

Physique

- *LRD (représentation)*
  - *représentation et stockage*

- *PDL (physical data language)*
  - *Create Index…*

Exemple : l'évaluation à l'UdeS

Le présent exemple reprend celui utilisé pour l'introduction à la théorie relationnelle.

Voir

- **TMR_01**

- Prédicats
- Données
- Types scalaires
- Tables (variables de relation)
- Diagramme
- Contraintes
- Variation
- Essai : données et tests

Les prédicats

- Activité :
  - L'activité identifiée par le sigle «sigle», décrite par le titre «titre», est offerte par l'UdeS.
- Étudiant :
  - L'étudiant identifié par le matricule «matricule», décrit par le nom «nom» et l'adresse «adresse» est admis à l'UdeS.
- TE :
  - Le type d'évaluation identifié par le code «code», décrit par la description «description» est autorisé à l'UdeS.
- Résultat :
  - Le résultat identifié par l'évaluation «TE» dans le cadre de l'activité «activité» au trimestre «trimestre», décrit par la note «note», a été obtenu par l'étudiant dont le matricule est «matricule».
  - L'étudiant dont le matricule est «matricule» est inscrit à l'activité «activité» au trimestre «trimestre» à l'UdeS.

Les données

Étudiant

matricule	nom	adresse
15113150	Paul	ᐳᕕᖏᓂᑲᖅ
15112354	Éliane	Blanc-Sablon
15113870	Mohamed	Tadoussac
15110132	Sergeï	Chandler

Activité

sigle	titre
IFT159	Analyse et programmation
IFT187	Éléments de bases de données
IMN117	Acquisition des médias numériques
IGE401	Gestion de projets
GMQ103	Géopositionnement

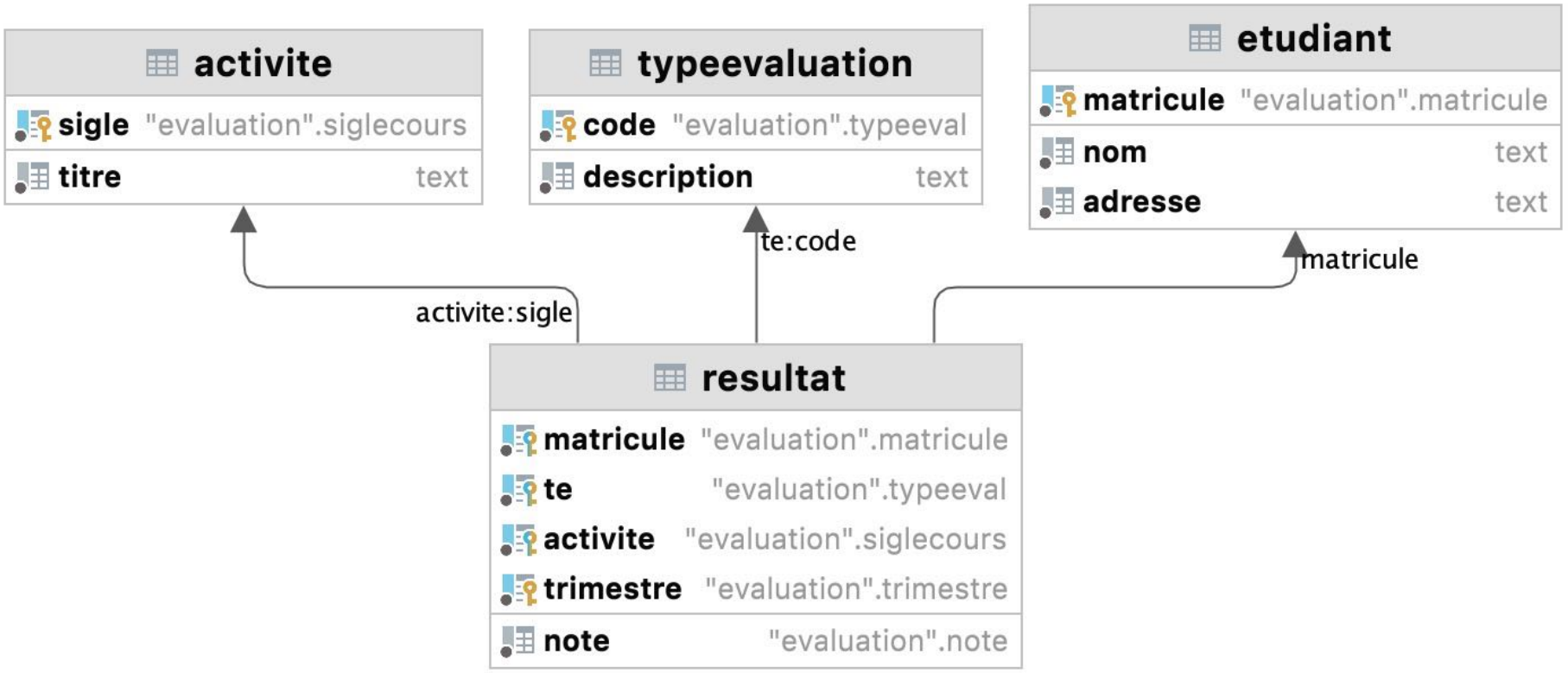
TypeÉvaluation

code	description
IN	Examen intra
FI	Examen final
TP	Travail pratique
PR	Projet

Résultat

matricule	TE	activité	trimestre	note
15113150	TP	IFT187	20133	80
15112354	FI	IFT187	20123	78
15113150	TP	IFT159	20133	75
15112354	FI	GMQ103	20123	85
15110132	IN	IMN117	20123	90
15110132	IN	IFT187	20133	45
15112354	FI	IFT159	20123	52

Le diagramme relationnel



Étape 1 - Définition des types scalaires (1)

SigleCours
{ x ∈ Texte | x = [A-Z]{3}[0-9]{3} }

```
CREATE DOMAIN SigleCours
  CHAR(6)
  CHECK(VALUE SIMILAR TO '[A-Z]{3}[0-9]{3}');
```

Matricule
{ x ∈ Texte | x = [0-9]{8} }

```
CREATE DOMAIN Matricule
  CHAR(8)
  CHECK(VALUE SIMILAR TO '[0-9]{8}');
```

TypeEval
{ x ∈ Texte | x = {TP, PR, IN, FI} }

```
CREATE DOMAIN TypeEval
  CHAR(2)
  CHECK(VALUE IN ('TP', 'PR', 'IN', 'FI'));
```

Étape 1 - Définition des types scalaires (2)

Note
{ x ∈ Entier | 0 <= x <= 100 }

```
CREATE DOMAIN Note
   SMALLINT
   CHECK (VALUE BETWEEN 0 AND 100);
```

Trimestre
{ t ∈ Texte | t = [0-9]{4}[1-3] }

```
CREATE DOMAIN Trimestre
   CHAR(5)
   CHECK(VALUE SIMILAR TO '[0-9]{4}[1-3]');
```

Étape 2 - Définition des tables (variables de relation) (1)

Activité {
  sigle : SigleCours ;
  titre : Texte
  }

Étudiant {
  matricule : Matricule ;
  nom : Texte ;
  adresse : Texte
  }

```
CREATE TABLE Activite (
    sigle         SigleCours NOT NULL,
    titre         Text NOT NULL
  );

CREATE TABLE Etudiant (
    matricule     Matricule NOT NULL,
    nom           Text NOT NULL,
    adresse       Texte NOT NULL
  );
```

Étape 2 - Définition des tables (variables de relation) (2)

TypeÉvaluation {
code : TypeEval ;
description : Text
}

```
CREATE TABLE TypeEvaluation (
    code          TypeEval NOT NULL,
    description   Text NOT NULL
  );
```

Résultat {
matricule : Matricule ;
TE : TypeEval ;
activité : SigleCours ;
trimestre : Trimestre ;
note : Note
}

```
CREATE TABLE Resultat (
    matricule     Matricule NOT NULL,
    TE            TypeEval NOT NULL,
    activite      SigleCours NOT NULL,
    trimestre     Trimestre NOT NULL,
    note          Note NOT NULL
  );
```

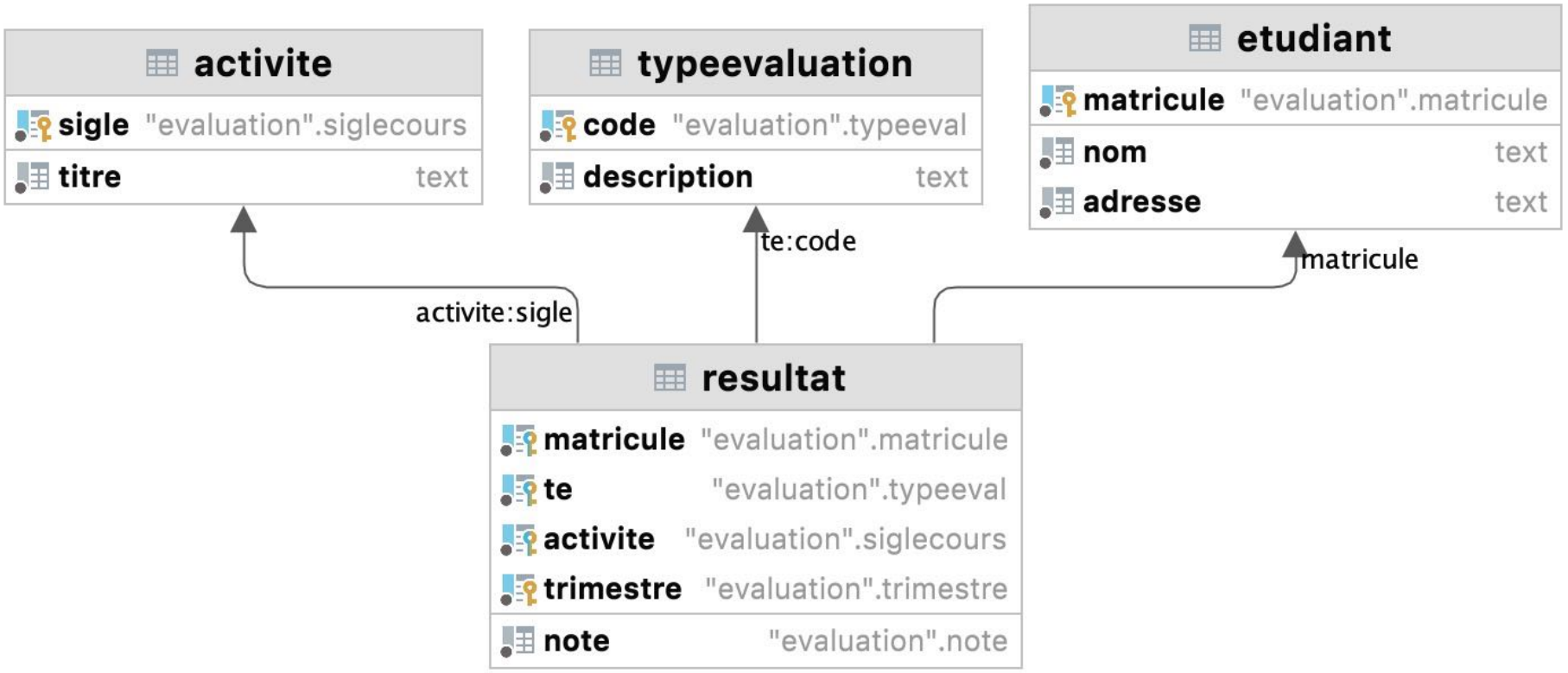
Étape 3 – Définition des contraintes (clés primaires)

```
CREATE TABLE Activite (
    sigle         SigleCours NOT NULL,
    titre         Text NOT NULL,
    PRIMARY KEY (sigle)
  );
CREATE TABLE Etudiant (
    matricule     Matricule NOT NULL,
    nom           Text NOT NULL,
    adresse       Text NOT NULL,
    PRIMARY KEY (matricule)
  );
CREATE TABLE TypeEvaluation (
    code          TypeEval NOT NULL,
    description   Text NOT NULL,
    PRIMARY KEY (code)
  );
```

Étape 3 – Définition des contraintes (clé primaire et clés référentielles)

```
CREATE TABLE Resultat (
  matricule    Matricule NOT NULL,
  TE           TypeEval NOT NULL,
  activite     SigleCours NOT NULL,
  trimestre    Trimestre NOT NULL,
  note         Note NOT NULL,
  PRIMARY KEY (matricule, activite, TE, trimestre),
  FOREIGN KEY (matricule)
    REFERENCES Etudiant (matricule),
  FOREIGN KEY (activite)
    REFERENCES Activite (sigle),
  FOREIGN KEY (TE)
    REFERENCES TypeEvaluation (code)
  );
```

Le diagramme relationnel (bis)



Variation – Associer un identifiant aux contraintes (1)

```
CREATE TABLE Activite (
    sigle         SigleCours NOT NULL,
    titre         Text NOT NULL,
    CONSTRAINT Activite_cc0 PRIMARY KEY (sigle)
  );
CREATE TABLE Etudiant (
    matricule     Matricule NOT NULL,
    nom           Text NOT NULL,
    adresse       Text NOT NULL,
    CONSTRAINT Etudiant_cc0 PRIMARY KEY (matricule)
  );
CREATE TABLE TypeEvaluation (
    code          TypeEval NOT NULL,
    description   Text NOT NULL,
    CONSTRAINT TypeEvaluation_cc0 PRIMARY KEY (code)
  );
```

Variation – Associer un identifiant aux contraintes (2)

```
CREATE TABLE Resultat (
  matricule    Matricule NOT NULL,
  TE           TypeEval NOT NULL,
  activite     CSibleCours NOT NULL,
  trimestre    Trimestre NOT NULL,
  note         Note NOT NULL,
  CONSTRAINT Resultat_cc0
    PRIMARY KEY (matricule, activite, TE, trimestre),
  CONSTRAINT Resultat_cr0 FOREIGN KEY (matricule)
    REFERENCES Etudiant (matricule),
  CONSTRAINT Resultat_cr1 FOREIGN KEY (activite)
    REFERENCES Activite (sigle),
  CONSTRAINT Resultat_cr2 FOREIGN KEY (TE)
    REFERENCES TypeEvaluation (code)
  );
```

Étape 4 – Préparation aux tests : exemple d'insertion de données

```
INSERT INTO Activite (sigle, titre) VALUES
    ('IFT159', 'Analyse et programmation'),
    ('IFT187', 'Éléments de bases de données'),
    ('IMN117', 'Acquisition des médias numériques'),
    ('IGE401', 'Gestion de projets'),
    ('GMQ103', 'Géopositionnement') ;
```

Étape 4 – Préparation aux tests : exemples de test

1. Quels sont les étudiants inscrits en IFT 187 ?
2. Quels sont les étudiants inscrits à une activité d'informatique à l'automne 2013 ?
3. Quels étaient les étudiants en situation d'échec au final à l'automne 2012 ?
4. Produire le relevé de notes d'Éliane.
5. Quels étudiants ne sont inscrits à aucune activité ?

R1 — Quels sont les étudiants inscrits en IFT 187 ?
Solution

- Clarification
  - La formulation utilisée pourrait indiquer qu'on s'intéresse au seul trimestre courant. Le requérant nous précise cependant qu'il vise toutes les inscriptions depuis la première offre de l'activité.
- Entête
  - InscritsIFT187 {matricule : Matricule}
- Requête
  - (Résultat σ (activité='IFT187')) π {matricule}

R1 — Quels sont les étudiants inscrits en IFT 187 ?
Script SQL

Rationnel :
(Résultat σ (activité='IFT187')) π {matricule}

SQL :

```
SELECT DISTINCT matricule
FROM Resultat
WHERE activite='IFT187'
```

- Rappels
  - La requête n'est correcte que si :
    - matricule est une clé référentielle de Résultat vers Étudiant ;
    - «on» définit qu'une inscription n'est en vigueur que si l'étudiant a complété une première évaluation et que celle-ci a été saisie ;
  - Si la première condition est naturelle, la deuxième l'est moins et soulève une interrogation quant à la justesse de la modélisation.

R2 — Quels sont les étudiants inscrits à une activité d'informatique à l'automne 2013 ? Solution

- Clarification
  - Une activité d'informatique est définie comme étant toute activité dont le sigle débute par le préfixe 'IFT'. Nous supposons qu'il existe une fonction 'préfixe' définie sur les valeurs de type Texte (couramment appelées 'chaines de caractères').
  - "Automne 2013" doit être recodé sous la forme '20133'.
- Entête
  - InscritsIFT {matricule : Matricule}
- Requête
  - (Résultat σ (préfixe(activité,3)='IFT') ∧ trimestre='20133') π {matricule}

R2 — Quels sont les étudiants inscrits à une activité d'informatique à l'automne 2013 ? Script SQL

Rationnel :

(Résultat σ (préfixe(activité,3)='IFT') ∧ trimestre='20133') π {matricule

SQL :

```
SELECT matricule
FROM Resultat
WHERE SUBSTRING(activite, 1, 3)='IFT'
  AND trimestre = '20133';
```

Avec doublons

matricule
15113150
15113150
15110132

```
SELECT DISTINCT matricule
FROM Resultat
WHERE SUBSTRING(activite, 1, 3)='IFT'
  AND trimestre = '20133';
```

Sans doublons

matricule
15113150
15110132

R3 — Quels étaient les étudiants en échec au final à l'automne 2012 ?
Solution

- Clarification
  - Une situation d'échec est une note inférieure à 60.
  - Un final est un 'Examen final' représenté par le code 'FI'.
  - «Automne 2012» doit être recodé sous la forme de l'entier 20123.
- Entête
  - Échecs20123 {matricule : Matricule}
- Requête
  - (Résultat σ (note<60 $\wedge$ TE='FI' $\wedge$ trimestre='20123'))π {matricule}

R3 — Quels étaient les étudiants en échec au final à l'automne 2012 ?
Script SQL

Rationnel :

(Résultat σ (note<60 ∧ TE='FI' ∧ trimestre='20123'))π {matricule}

SQL :

```
SELECT DISTINCT matricule
FROM Resultat
WHERE note < 60
  AND TE = 'FI'
  AND trimestre = '20123';
```

R4 — Produire le relevé de notes d'Éliane – variante 1.
Solution

- Clarification
  - Le matricule a été introduit pour différencier les homonymes. Un relevé produit sur la seule base du nom est donc susceptible d'être inexact. En conséquence, **nous demanderons à Éliane** son matricule (nous ne consultons pas la base de données).
- Entête
  - RelevéÉliane
    { TE : TypeEval ; activité : SigleCours ;
    trimestre : Trimestre ; note : Note }
- Requête
  - (Résultat σ (matricule='15112354')) π {TE, activité, trimestre, note}

R4 — Produire le relevé de notes d'Éliane – variante 1.
Script SQL

Rationnel :

(Résultat σ (matricule='15112354')) π {TE, activité, trimestre, note}

SQL :

```
SELECT DISTINCT TE, activite, trimestre, note
FROM Resultat
WHERE matricule = '15112354' ;
```

Pourquoi n'est-ce pas nécessaire, mais néanmoins pas inexact, de spécifier DISTINCT ?

R4 — Produire le relevé de notes d'Éliane – variante 2.
Solution

- Clarification
  - On demande d'ajouter le titre de l'activité dans le relevé.
- Entête
  - Relevé2Éliane {TE : Texte ; *sigle : Texte ;* titre : Texte ; trimestre : Entier ; note : Entier [0..100]}
- Requête
  - Remarquons qu'il est nécessaire de joindre la relation Activité pour obtenir le titre et que l'attribut de jointure n'y porte pas le même nom que dans la relation Résultat.
  - (((Résultat σ (matricule='15112354'))
    ρ {activité → sigle})
    ⋈ Activité)
    π {TE, *sigle*, titre, trimestre, note}

R4 — Produire le relevé de notes d'Éliane – variante 2.
Script SQL

Rationnel :

(((Résultat σ (matricule='15112354'))
ρ {activité → sigle})
⋈ Activité)
π {TE, *sigle*, titre, trimestre, note}

SQL :

```
SELECT TE, sigle, titre, trimestre, note
FROM Resultat
        JOIN Activite ON (activite = sigle)
WHERE matricule = '15112354';
```

R5 — Quels étudiants ne sont inscrits à aucune activité ?
Analyse

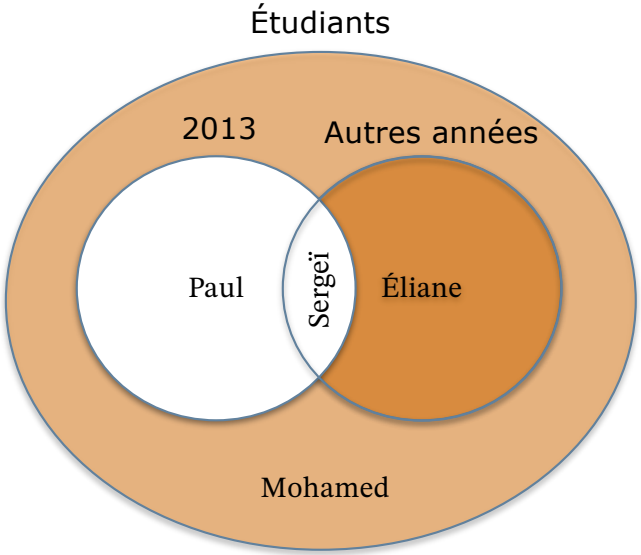
- Clarification
  - Encore une fois, il est nécessaire de faire préciser la période à couvrir, en termes de trimestres. Supposons que ce soit les trois trimestres de l'année 2013, la question devient donc :
    - Quels étudiants ne sont inscrits à aucune activité en 2013 ?
  - Supposons également qu'on désire avoir un maximum d'information sur ces étudiants et pas seulement leur matricule (à savoir tous les attributs disponibles dans la relation Étudiant).
- Entête
  - NonInscrits2013
    {matricule : Matricule ; nom : Texte ; adresse : Texte}

R5 — Quels étudiants ne sont inscrits à aucune activité ?
Solution

- Requête
  - Nous réalisons cette requête en calculant d'abord les étudiants inscrits à au moins une activité en 2013 puis en les soustrayant de l'ensemble des étudiants.
  - Inscrits2013 :
    (Résultat σ ('20131'≤trimestre≤'20133')) π {matricule}
  - NonInscrits2013 :
    Étudiant – (Étudiant $\bowtie$ Inscrits2013)
  - ou, en extension :
    Étudiant –
    (Étudiant $\bowtie$
    ((Résultat σ ('20131'≤trimestre≤'20133')) π {matricule}))

R5 — Quels étudiants ne sont inscrits à aucune activité ?
Script SQL, version 1

```
SELECT matricule, nom, adresse
FROM Etudiant
EXCEPT
  SELECT matricule, nom, adresse
  FROM Etudiant
    JOIN (
          SELECT matricule
          FROM Resultat
          WHERE '20131' <= trimestre
            AND trimestre <= '20133'
        ) AS Inscrit2013
    USING(matricule);
```



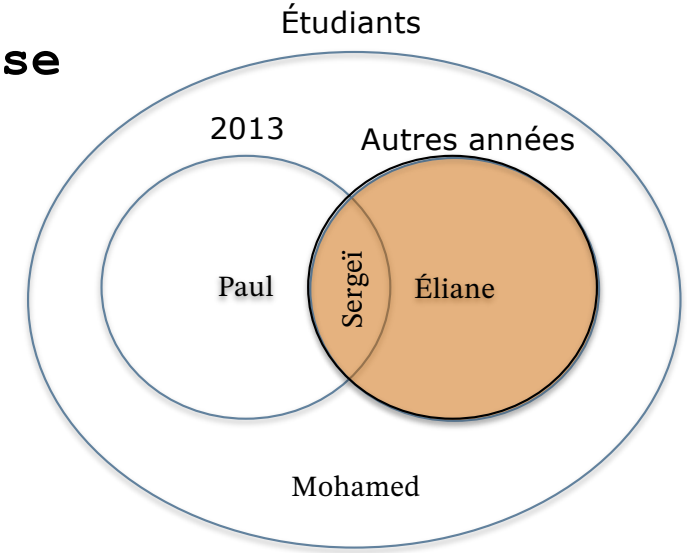
matricule	nom	adresse
15112354	Éliane	Blanc-Sablon
15113870	Mohamed	Tadoussac

R5 — Quels étudiants ne sont inscrits à aucune activité ?
Script SQL, correction

- Justin remarque qu'il est possible de simplifier l'expression précédente si on admet qu'au sein du modèle la négation de la proposition
  (20131≤trimestre≤20133)
  s'exprime par
  (trimestre<20131 ⋁ 20133<trimestre)
- Il obtient alors
  (Étudiant
  ⋈(Résultat σ (trimestre<'20131' ⋁ '20133'<trimestre)))
  π {matricule, nom, adresse}
- A-t-il raison ?
  - NON !

R5 — Quels étudiants ne sont inscrits à aucune activité ?
Script SQL, version 1

```
SELECT DISTINCT matricule, nom, adresse
FROM Etudiant
  JOIN (
       SELECT matricule
       FROM Resultat
       WHERE trimestre < '20131'
          OR '20133' > trimestre
     ) AS NonInscrit2013
  USING(matricule);
```



matricule	nom	adresse
15112354	Éliane	Blanc-Sablon
15110132	Sergeï	Chandler

R5 — Quels étudiants ne sont inscrits à aucune activité ? Pourquoi cette incohérence ?

- Parce qu'un étudiant peut s'inscrire lors de plusieurs trimestres, on ne peut pas logiquement conclure qu'un étudiant inscrit en 2013 ne s'est inscrit à aucune autre année :
  - Par exemple, Sergeï s'est inscrit en 2012 et en 2013.
- On ne peut pas logiquement conclure qu'un étudiant qui ne s'est pas inscrit en 2013 s'est inscrit lors d'une autre année :
  - Par exemple, Mohamed est étudiant, mais ne s'est encore inscrit à aucune activité.

Revue

Revoir certaines sections de la présentation

TRM_01-Fondements_TD

- La démarche
  - Élaboration du MLD (entêtes, prédicats, clés)
  - Élaboration d'une requête
  - Élaboration des cas de test
- Les règles de pratique
- Les colles du prof
  - Anecdote permettant de relativiser le terme *Big Data*
  - Ou comment IBM estimait en 1956 que le marché pour dispositif de stockage de 4 Mo était mondialement limité à au plus 1000 unités.

Les colles du prof

- Quelle est la différence entre un modèle logique de données, un diagramme et un script ?
- Y a-t-il une seule façon de traduire une expression relationnelle en script SQL ?
- Comment décririez-vous l'instruction SELECT ?
- Est-il pertinent de formuler une requête d'abord sous forme d'expression relationnelle puis de la traduire en script SQL ?
- Est-il préférable de formuler directement une requête en script SQL ?
- Comparer les prédicats utilisés dans BD100 avec ceux proposés de l'exemple *Evaluation* déposé dans le répertoire public du cours. Décrire la portée des différences, indiquer les "meilleures" formulations et motiver les choix.