

- Les jointures se font le long des colonnes étrangères.
- Préférence pour JOIN ON au JOIN USING ("attribut")  
 ↳ pas de colonne en double
- sqlite 3 :
  - \* Readers on → les en-têtes
  - \* mode column → affichage en colonne
- NATURAL JOIN fait la jointure sur tous les attributs qui partagent le même nom
- \* Requêtes imbriquées à éviter en général :
  - \* Par exemple pour tout service sur les étudiants qui sont dans inscrits :
    - avec une jointure :
- SELECT Etudiant.\* from Etudiant INNER

WHERE numEt IN (select numet  
from Inscrit)

```
0          0          0          SEARCH TABLE Etudiant USING INTEGER PRIMAR
Y KEY (rowid=?) 0          0          EXECUTE LIST SUBQUERY 1
1          0          0          SCAN TABLE Inscrit

sqlite> select Etudiant.* from Etudiant WHERE etudiant.numet IN (select numet
from Inscrit) ;
NumEt      NomEt      Adresse
-----  -----
1111       Armand A.  1234 rue premiere
1112       Berthe B.  2345 rue deuxieme
1114       David D.  4567 rue quatriem
1115       Erwan E.  5678 rue cinquiem
1116       Fabien F. 6789 rue sixieme
1117       Gerald G. 7890 rue septieme
1118       Herbert H. 8901 rue huitieme
1119       Jacques J. 9012 rue neuvieme
sqlite> select Etudiant.* from Etudiant WHERE etudiant.numet = (select numet f
rom etudiant where nomet = 'Albert A.' ) ;
sqlite> select Etudiant.* from Etudiant WHERE etudiant.numet = (select numet f
rom etudiant where nomet = 'Armand A.' ) ;
NumEt      NomEt      Adresse
-----  -----
1111       Armand A.  1234 rue premiere
sqlite>
```

Requête imbriquée :

- dans un From OK

- via intermédiaire OK (requête nommée)

- ici on change le type de données, on fait l'égalité entre un scalaire etudiant.numet et un vecteur (retourné par la requête imbriquée) => si on

requête un homonyme Armand.A. on aura un problème :  
↳ il faudrait remplacer = par IN

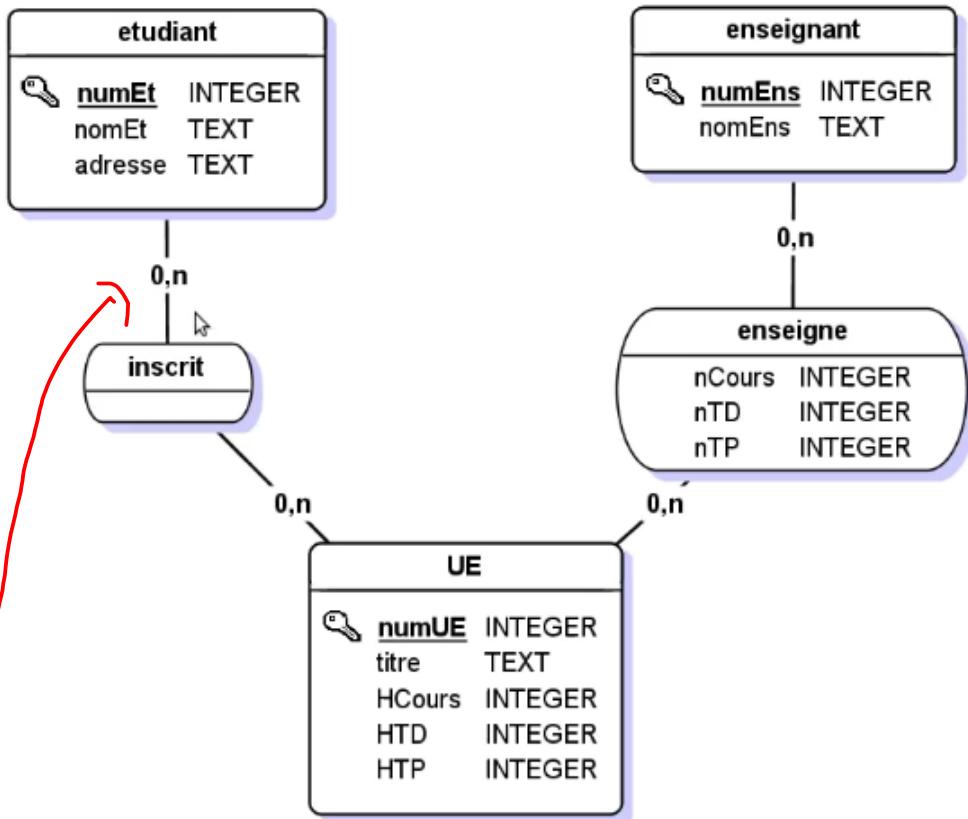
```
sqlite> select Etudiant.* from Etudiant WHERE (etudiant.numet, etudiant.nomet)
    IN (select numet,nomet from etudiant where nomet = 'Armand A.' );
NumEt      NomEt      Adresse
-----  -----  -----
1111      Armand A.  1234 rue premiere
sqlite>
```

Si les types de la condition du WHERE sont cohérents

```
**** Armand A. 1234 rue premiere
sqlite> select * from (select * from etudiant where nomet like 'A%') where num
et > 2000;
sqlite> select * from (select * from etudiant where nomet like 'A%') where num
et > 1000;
NumEt      NomEt      Adresse
-----  -----  -----
1111      Armand A.  1234 rue premiere
sqlite>
```

Requêtes imbriquées OK ..  
dans un from

- Limitations de SQLite => certaines fonctions comme ALL ne sont pas implementées



Si un étudiant peut s'inscrire à une seule UE, la table inscrit n'a plus de sens.

\* Vocabulaire:

- tuple => record => enregistrement

\* les clefs étrangères ne font pas vraiment partie de la normalisation

↳ elles servent à consolider les liens entre les tables

- \* Renommage => indispensable en cas d'autojoin
- \* Peux sqlite3 ;  
pour lire un fichier de script SQL

```

sqlite> .read test.sql
NumEt      NomEt      Adresse
-----
1111      Armand A.  1234 rue premiere
1112      Berthe B.  2345 rue deuxieme
1113      Cendrine C 3456 rue troisiem
1114      David D.  4567 rue quatriem
1115      Erwan E.  5678 rue cinquiem
1116      Fabien F. 6789 rue sixieme
1117      Gerald G. 7890 rue septieme
1118      Herbert H. 8901 rue huitieme
1119      Jacques J. 9012 rue neuvieme
sqlite> .read test.sql
NumEt      NomEt      Adresse
-----
1116      Fabien F.  6789 rue sixieme
1117      Gerald G.  7890 rue septiem
1118      Herbert H. 8901 rue huitiem
1119      Jacques J. 9012 rue neuviem

```

- \* Logiciels :
  - SQLite DB Browser => Léger / sans dépendance
  - DBBeaver => Logiciel plus complet

# \* SQL :

- insensible à la cause  
sur les mots clefs

- il faut insister !

# Les rives : Vue extérieure

Per se dey  
parentheses

The screenshot shows a Linux desktop environment with several windows open:

- Terminal:** Shows a SQLite session with multiple queries. The first query reads from a file named `test.sql`. The second query reads from a view named `etu2`. Both queries involve tables `Inseignant`, `Enseigne`, `Etudiant`, and `Inscrit`.
- File Browser:** A sidebar showing the directory structure. It includes files like `base-etudiant.sql`, `base-stanford.sql`, `README.md`, `READEMD`, `sql-murder-mystery.db`, and `test.sql`.
- SQL Editor:** A window titled "Viewing Romuald Thion's screen" containing the contents of `test.sql`. The SQL code creates a view `Etu2` that joins `Etudiant` and `Inscrit` and groups by `Etudiant.nume`.

```
116 Fabien F. 6789 rue sixieme
117 Gerald G. 7890 rue septieme
118 Herbert H. 8901 rue huitieme
119 Jacques J. 9012 rue neuvieme
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite> .read test.sql
sqlite> .tables
Inseignant Enseigne Etu2      Etudiant    Inscrit     UE
sqlite> select * from etu2;
numEt NomEt      Adresse      count(numUE)
-----
111  Armand A.  1234 rue premiere 3
112  Berthe B.  2345 rue deuxieme 2
114  David D.   4567 rue quatriem 4
115  Erwan E.   5678 rue cinquiem 1
116  Fabien F.  6789 rue sixieme 2
117  Gerald G. 7890 rue septieme 4
118  Herbert H. 8901 rue huitieme 2
119  Jacques J. 9012 rue neuvieme 3
sqlite> .read test.sql
sqlite> .tables
Inseignant Enseigne Etu2      Etudiant    Inscrit     UE
sqlite> select * from etu2;
numEt NomEt      count(numUE)
-----
111  Armand A.  3
112  Berthe B.  2
114  David D.   4
115  Erwan E.   1
116  Fabien F.  2
117  Gerald G.  4
118  Herbert H. 2
119  Jacques J.  3
```

```
1 DROP VIEW IF EXISTS Etu2;
2
3 CREATE VIEW Etu2 AS
4 SELECT Etudiant.Nume, Etudiant.nomEt, count(numUE)
5 FROM Etudiant JOIN Inscrit USING (Nume)
6 GROUP BY Etudiant.nume
7 ;
```

Vue locale avec WITH (crée un alias)

pour des requêtes imbuvables

```
base-etudiant.sql x
1 WITH Etu2 AS(
2   . . .SELECT Etudiant.Numeret, Etudiant.nomet, count(numUE)
3   . . .FROM Etudiant LEFT JOIN Inscrit USING (NumEt)
4   . . .GROUP BY Etudiant.numret)
5 |
6 SELECT *
7 FROM Etu2;
```

\* Les vues sont intéressantes pour les applis web

\* les vues persistent à la fermeture de la table (différence avec les WITH)

↳ si c'est juste pour une requête, mieux valoir utiliser WITH.

```
Activities Text Editor Terminal Viewing Romuald Thion's screen /Documents/Enseignement/DBI_EB/block4_du_vendredi_17_mars_2017/bases_de donnees/introduction/TP base-etudiant.sql *test.sql
1 --SELECT *
2 --FROM (SELECT * FROM Etudiant WHERE Numet > 1114);
3
4
5
6
7
8 WITH Tmp AS(
9   SELECT * FROM Etudiant WHERE Numet > 1114)
10
11 SELECT *
12 FROM tmp;
```

```
Gerald G. Gerald G.
Gerald G. Herbert H.
Gerald G. Jacques J.
Herbert H. Armand A.
Herbert H. Berthe B.
Herbert H. Cendrine C
Herbert H. David D.
Herbert H. Erwan E.
Herbert H. Fabien F.
Herbert H. Gerald G.
Herbert H. Herbert H.
Herbert H. Jacques J.
Jacques J. Armand A.
Jacques J. Berthe B.
Jacques J. Cendrine C
Jacques J. David D.
Jacques J. Erwan E.
Jacques J. Fabien F.
Jacques J. Gerald G.
Jacques J. Herbert H.
Jacques J. Jacques J.
sqlite> .read test.sql
sqlite> .tables
Enseignant Enseignant Etudiant Inscrit UE test
sqlite> .read test.sql
1115|Erwan E.|5678 rue cinquième
1116|Fabien F.|6789 rue sixième
1117|Gerald G.|7890 rue septième
1118|Herbert H.|8901 rue huitième
1119|Jacques J.|9012 rue neuvième
sqlite> .read test.sql
1115|Erwan E.|5678 rue cinquième
1116|Fabien F.|6789 rue sixième
1117|Gerald G.|7890 rue septième
1118|Herbert H.|8901 rue huitième
1119|Jacques J.|9012 rue neuvième
sqlite> |
```

\* Suppression de table/vue

un autre exemple

A screenshot of a terminal window titled "Viewing Romuald Thion's screen". The terminal shows the following SQLite session:

```
sqlite> .mode column
sqlite> :headers on
sqlite> .read test.sql
NumEt      NomEt      count(numUE)
-----      -----      -----
1111      Armand A.    3
1112      Berthe B.    2
1113      Cendrine C.  8
1114      David D.    4
1115      Erwan E.     1
1116      Fabien F.    2
1117      Gerald G.   4
1118      Herbert H.   2
1119      Jacques J.   3
sqlite> .read test.sql
NumEt      NomEt      Nb_UE
-----      -----      -----
1111      Armand A.    3
1112      Berthe B.    2
1113      Cendrine C.  0
1114      David D.    4
1115      Erwan E.     1
1116      Fabien F.    2
1117      Gerald G.   4
1118      Herbert H.   2
1119      Jacques J.   3
sqlite> .read test.sql
Error: near line 1: no such column: Etudiant.Numet
sqlite> .read test.sql
Error: near line 1: no such column: Etudiant.Numet
sqlite> .read test.sql
NumEt      NomEt      Nb_UE
-----      -----      -----
1113      Cendrine C.  0
1114      David D.    4
1115      Erwan E.     1
1116      Fabien F.    2
1117      Gerald G.   4
1118      Herbert H.   2
1119      Jacques J.   3
sqlite>
```

The terminal also contains a copy of the "test.sql" file:

```
-- SELECT E.Numet, E.nomet, count(numUE) as Nb_UE
-- FROM Etudiant E LEFT JOIN Inscrit I USING (NumEt)
-- WHERE E.numet > 1112
-- GROUP BY E.numet
```

\* indispensable pour les auto-jointures  
petit hack : on utilise une condition sur l'adjonction - graphique

A screenshot of a terminal window titled "Viewing Romuald Thion's screen". The terminal shows the following SQLite session:

```
rwan E.  Fabien F.
rwan E.  Gerald G.
rwan E.  Herbert H.
rwan E.  Jacques J.
abien F. Armand A.
abien F. Berthe B.
abien F. Cendrine C
abien F. David D.
abien F. Erwan E.
abien F. Fabien F.
abien F. Gerald G.
abien F. Herbert H.
abien F. Jacques J.
erald G. Armand A.
erald G. Berthe B.
erald G. Cendrine C
erald G. David D.
erald G. Erwan E.
erald G. Fabien F.
erald G. Gerald G.
erald G. Herbert H.
erald G. Jacques J.
erbert H. Armand A.
erbert H. Berthe B.
erbert H. Cendrine C
erbert H. David D.
erbert H. Erwan E.
erbert H. Fabien F.
erbert H. Gerald G.
erbert H. Herbert H.
erbert H. Jacques J.
acques J. Armand A.
acques J. Berthe B.
acques J. Cendrine C
acques J. David D.
acques J. Erwan E.
acques J. Fabien F.
acques J. Gerald G.
acques J. Herbert H.
acques J. Jacques J.
sqlite> .read test.sql
```

The terminal also contains a copy of the "test.sql" file:

```
-- SELECT E.Numet, E.nomet, count(numUE) as Nb_UE
-- FROM Etudiant E LEFT JOIN Inscrit I USING (NumEt)
-- WHERE E.numet > 1112
-- GROUP BY E.numet

SELECT E1.nomEt, E2.nomEt
FROM Etudiant E1, Etudiant E2
WHERE E1.nomEt < E2.nomEt;
```

Handwritten notes on the right side of the terminal window:

- que
- pour
- suppli
- me
- les
- doubles
- ↓
- classique
- des
- exams /

## Partie des alias :

Cela peut dépendre des

The screenshot shows a terminal window titled "Viewing Romuald Thion's screen". It displays several SQLite commands and their results:

```
base-etudiant.sql | test.sql
1 . SELECT E.NumeT, E.numeT AS n, COUNT(numUE) AS Nb_UE
2 FROM Etudiant E LEFT JOIN Inscrit I USING (numET)
3 WHERE E.numeT > 1112 AND I.date like '08'
4 GROUP BY E.numeT
5
6
7 --SELECT E1.numeT, E2.numeT
8 --FROM Etudiant E1, Etudiant E2
9 --WHERE E1.numeT < E2.numeT;

SELECT NomEt, Nb_UE
-----
113 Cendrine C. 0
114 David D. 4
115 Erwan E. 1
116 Fabien F. 2
117 Gerald G. 4
118 Herbert H. 2
119 Jacques J. 3

SELECT NomEt, Nb_UE
-----
114 David D. 4
116 Fabien F. 2
117 Gerald G. 4
118 Herbert H. 2
119 Jacques J. 3

SELECT NomEt, Nb_UE
-----
117 Gerald G. 4

SELECT n, Nb_UE
-----
117 Gerald G. 4
```

\* Utilité de LEFT JOIN :

↳ si on veut garder tous ces enregistrements de la table de gauche même si'ils ne sont pas dans le jointure

↳ chaque fois qu'on demande un tableau de synthèse par exemple le nombre de UE où sont inscrits les

→ étudiants => on veut les étudiants qui ne sont pas inscrits à une UE

The screenshot shows a terminal window with the following content:

```
Activities Text Editor Terminal
Viewing Romuald Thion's screen
File Edit View Search Terminal Help
NumEt n Nb_UE
1117 Gerald G. 4
sqlite> .read test.sql
NumEt n Nb_UE
1111 Armand A. 3
1112 Berthe B. 2
1113 Cendrine C 0
1114 David D. 4
1115 Erwan E. 1
1116 Fabien F. 2
1117 Gerald G. 4
1118 Herbert H. 2
1119 Jacques J. 3
sqlite> .read test.sql
NumEt NomEt Adresse NumUE
1111 Armand A. 1234 rue premiere 3
1111 Armand A. 1234 rue premiere 4
1111 Armand A. 1234 rue premiere 5
1112 Berthe B. 2345 rue deuxieme 4
1112 Berthe B. 2345 rue deuxieme 5
1113 Cendrine C 3456 rue troisiem
1114 David D. 4567 rue quatriem 1
1114 David D. 4567 rue quatriem 2
1114 David D. 4567 rue quatriem 3
1114 David D. 4567 rue quatriem 6
1115 Erwan E. 5678 rue cinquiem 4
1116 Fabien F. 6789 rue sixieme 5
1116 Fabien F. 6789 rue sixieme 6
1117 Gerald G. 7890 rue septieme 1
1117 Gerald G. 7890 rue septieme 2
1117 Gerald G. 7890 rue septieme 3
1117 Gerald G. 7890 rue septieme 4
1118 Herbert H. 8901 rue huitieme 5
1118 Herbert H. 8901 rue huitieme 6
1119 Jacques J. 9012 rue neuvieme 1
1119 Jacques J. 9012 rue neuvieme 2
1119 Jacques J. 9012 rue neuvieme 3
sqlite>
```

The terminal window also displays the following SQL code:

```
base-etudiant.sql
1 --SELECT * --E.NumEt, E.nomEt as n, count(numUE) as Nb_UE
2 --FROM Etudiant E LEFT JOIN Inscrit I USING (NumEt)
3
4
5
6 --SELECT E1.nomEt, E2.nomEt
7 --FROM Etudiant E1, Etudiant E2
8 --WHERE E1.nomEt < E2.nomEt;
```

- \* SQLite est assez permisif.
  - ↳ pour le produit cartésien il utilise JOIN au lieu de CROSS JOIN
- \* Pour chaque Haus, sélectionner le nombre d'UE qui ont ce nombre d'Haus
  - ↳ par exemple SQLite accepte de mettre dans un agrégat un attribut pas dans

le GROUP BY => dommage qu'il n'y ait pas de mode strict

The screenshot shows a terminal window with the following content:

```
Activities Terminal
File Edit View Search Terminal Help
Viewing Romuald Thion's screen
2.5GHz test.sql test.sql
base-etudiant.sql x | x
1 -- SELECT E.NumEt, E.nomEt as n, count(numUE) as Nb_UE
2 -- FROM Etudiant E LEFT JOIN Inscrit I USING (NumEt)
3
4
5
6 SELECT HCours, titre, count(numUE)
7 FROM UE
8 GROUP BY HCours
9 ORDER BY HCours DESC;
118    Herbert H. 8901 rue huitieme
119    Jacques J. 9812 rue neuvieme
sqlite> select * from UE;
numUE      Titre      HCours      HTD      HTP
-----      -----      -----      -----      -----
        Analyse    20.0      25.0      0.0
        Algebre    20.0      25.0      0.0
        Programmat 15.0      15.0      15.0
        Algorithmique 20.0      15.0      15.0
        Bases de d 18.0      18.0      18.0
        Reseaux     6.0       0.0       2.0
sqlite> .read test.sql
HCours      count(numUE)
-----      -----
  0.0          1
  5.0          1
  8.0          1
  9.0          3
sqlite> .read test.sql
HCours      count(numUE)
-----      -----
  0.0          3
  8.0          1
  5.0          1
  0.0          1
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite>
sqlite> .read test.sql
HCours      Titre      count(numUE)
-----      -----      -----
  0.0      Algorithmique 3
  8.0      Bases de donn 1
  5.0      Programmation 1
  0.0      Reseaux      1
sqlite>
```

SQLite sont des valeurs aléatoires  
aucune garantie que ce soit  
déterministe (si les insert n'ont  
pas été faits dans le m<sup>e</sup> ordre)

\* Dans SQL, si on ne met  
pas de ORDER BY on n'a aucune  
garantie sur l'ordre  
> si on a une Primary Key,  
on a un index et il est possible  
que le SGBD s'appuie dessus

\* Méthodologie pour écrire une requête :

- il n'en existe pas vraiment
- il faut bien connaître le schéma

\* On peut mettre des constantes dans les requêtes :

```
sqlite> SELECT 1 FROM Etudiant;
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
1
```

```
sqlite> SELECT Numet, 1 FROM Etudiant;  
1111      1  
1112      1  
1113      1  
1114      1  
1115      1  
1116      1  
1117      1  
1118      1  
1119      1  
sqlite> SELECT Numet, 1 FROM Etudiant.
```

\* On peut se servir de SQL comme d'une calculatrice

```
sqlite> SELECT (1+2);  
3
```

\* LEFT JOIN

↳ NATURAL JOIN

ou INNER

UNION

tous les enregistrements de l'ensemble de données de la table de gauche qui ont une correspondance dans la table de droite

## \* Commandes de modification

\* ALTER TABLE => changer la structure d'une table

```
sqlite> ALTER TABLE Etudiant ADD COLUMN Age Integer;
sqlite> .tables
Enseignant  Enseigne    Etudiant   Inscrit    UE      test
sqlite> select * from Etudiant ;
1111        Armand A.   1234 rue premiere
1112        Berthe B.   2345 rue deuxieme
1113        Cendrine C  3456 rue troisiem
1114        David D.   4567 rue quatriem
1115        Erwan E.   5678 rue cinquiem
1116        Fabien F.  6789 rue sixieme
1117        Gerald G.  7890 rue septieme
1118        Herbert H. 8901 rue huitieme
1119        Jacques J. 9012 rue neuvieme
sqlite> .headers on
sqlite> select * from Etudiant ;
NumEt      NomEt       Adresse      Age
-----      -----
1111        Armand A.  1234 rue premiere
1112        Berthe B.  2345 rue deuxieme
1113        Cendrine C 3456 rue troisiem
1114        David D.  4567 rue quatriem
1115        Erwan E.  5678 rue cinquiem
1116        Fabien F. 6789 rue sixieme
1117        Gerald G. 7890 rue septieme
1118        Herbert H. 8901 rue huitieme
1119        Jacques J. 9012 rue neuvieme
sqlite> UPDATE Etudiant SET age = 18 WHERE Numet < 1115;
```

\* UPDATE ... SET ... WHERE ...

↳ mis à jour d'une ligne

```
sqlite> UPDATE Etudiant SET age = 18 where Numet < 1115;
sqlite> select * from Etudiant ;
I
NumEt      NomEt       Adresse      Age
-----      -----
1111        Armand A.  1234 rue premiere 18
1112        Berthe B.  2345 rue deuxieme 18
1113        Cendrine C 3456 rue troisiem 18
1114        David D.  4567 rue quatriem 18
1115        Erwan E.  5678 rue cinquiem
1116        Fabien F. 6789 rue sixieme
1117        Gerald G. 7890 rue septieme
1118        Herbert H. 8901 rue huitieme
1119        Jacques J. 9012 rue neuvieme
```

\* Pas de boucle en SQL:

↳ existe dans des extensions  
de SQL

↳ PL/SQL

Procedural language SQL

\* Modélisat.

↳ plusieurs  
façons possibles

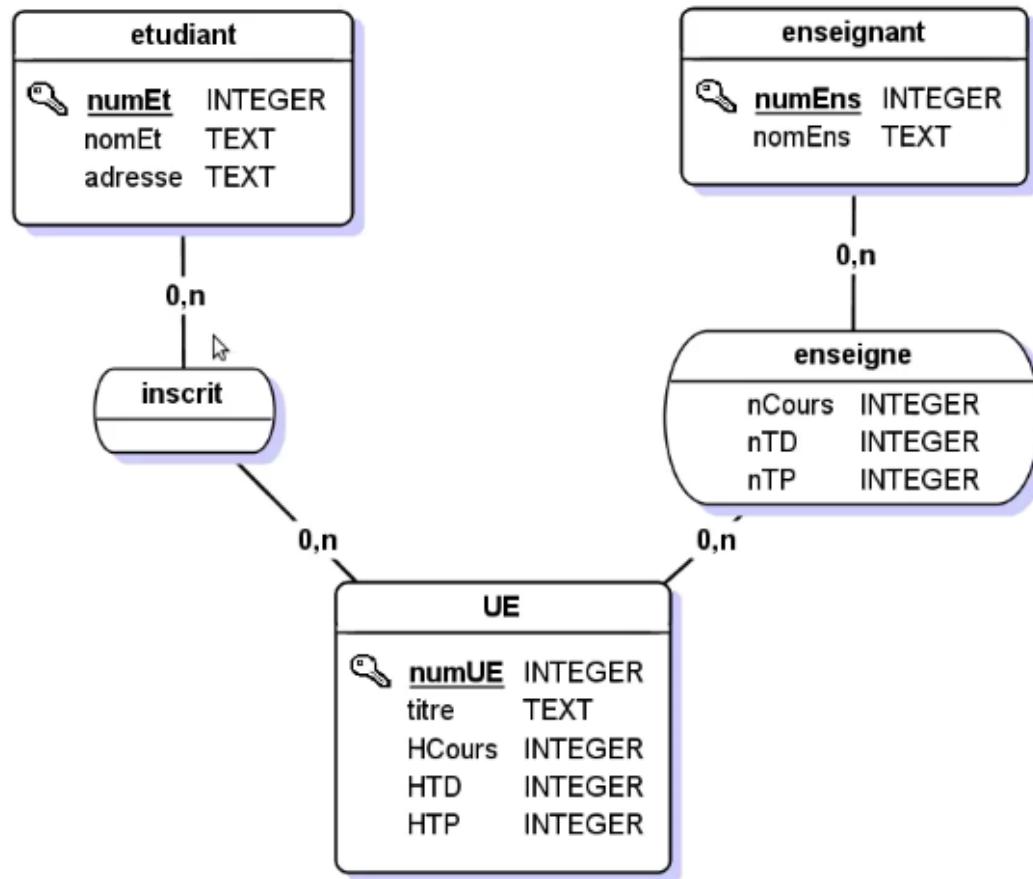
↳ on peut toujours vérifier  
que certaines contraintes  
sont respectées :

→ un enseignant peut-il  
enseigner plusieurs UE?

→ un étudiant <sup>peut-il avoir</sup> plusieurs  
adresses!



\* Reconstruire la table universelle à partir du schéma :



avec des jointures.

NumEt	NomEt	Adresse	Age	NumUE	Titre	HCours	HTD	HTP
1111	Armand A.	1234 rue premiere	18	5	Bases de donnees	18.0	18.0	18.0
1111	Armand A.	1234 rue premiere	18	4	Algorithmique	20.0	15.0	15.0
1111	Armand A.	1234 rue premiere	18	3	Programmation	15.0	15.0	15.0
1112	Berthe B.	2345 rue deuxieme	18	4	Algorithmique	20.0	15.0	15.0
1112	Berthe B.	2345 rue deuxieme	18	5	Bases de donnees	18.0	18.0	18.0
1114	David D.	4567 rue quatriem	18	6	Reseaux	6.0	0.0	2.0
1114	David D.	4567 rue quatriem	18	1	Analyse	20.0	25.0	0.0
1114	David D.	4567 rue quatriem	18	2	Algebre	20.0	25.0	0.0
1114	David D.	4567 rue quatriem	18	3	Programmation	15.0	15.0	15.0
1115	Erwan E.	5678 rue cinquiem	18	4	Algorithmique	20.0	15.0	15.0
1116	Fabien F.	6789 rue sixieme	18	5	Bases de donnees	18.0	18.0	18.0
1116	Fabien F.	6789 rue sixieme	18	6	Reseaux	6.0	0.0	2.0
1117	Gerald G.	7890 rue septieme	18	1	Analyse	20.0	25.0	0.0
1117	Gerald G.	7890 rue septieme	18	2	Algebre	20.0	25.0	0.0
1117	Gerald G.	7890 rue septieme	18	3	Programmation	15.0	15.0	15.0
1117	Gerald G.	7890 rue septieme	18	4	Algorithmique	20.0	15.0	15.0
1118	Herbert H.	8901 rue huitieme	18	5	Bases de donnees	18.0	18.0	18.0
1118	Herbert H.	8901 rue huitieme	18	6	Reseaux	6.0	0.0	2.0
1119	Jacques J.	9012 rue neuvieme	18	1	Analyse	20.0	25.0	0.0
1119	Jacques J.	9012 rue neuvieme	18	2	Algebre	20.0	25.0	0.0
1119	Jacques J.	9012 rue neuvieme	18	3	Programmation	15.0	15.0	15.0

Table universelle  $\Rightarrow$  intérêt  
l'héritage  $\Rightarrow$  feuille de tableau  
avec toutes les données

$\hookrightarrow$  On travaille notamment  
l'autre sens (**Normalisation**)

$\Rightarrow$  on peut découper, stocker  
séparément

$\hookrightarrow$  éviter les anomalies  
d'insertion, de suppression

\* Open data  $\Rightarrow$  souvent données  
dans une seule table

\* La table universelle pose  
problème si on modifie les  
données.

. Il y a beaucoup de répétitions

$\hookrightarrow$  on voudrait ne pas retrouver les  
mêmes des candidats dans le fichier  
des résultats aux élections présidentielles  
de 2017 par exemple

