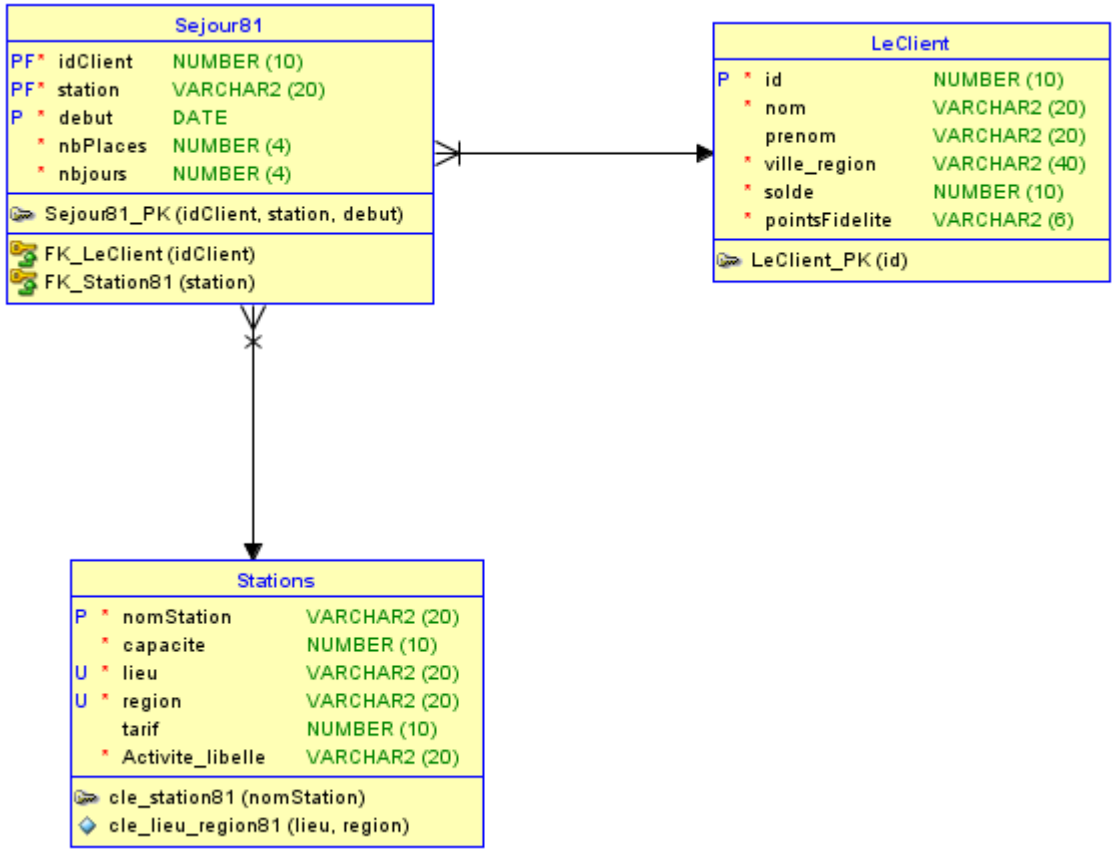
Flux d’extraction

Giuseppe Berio

2023

Rappel

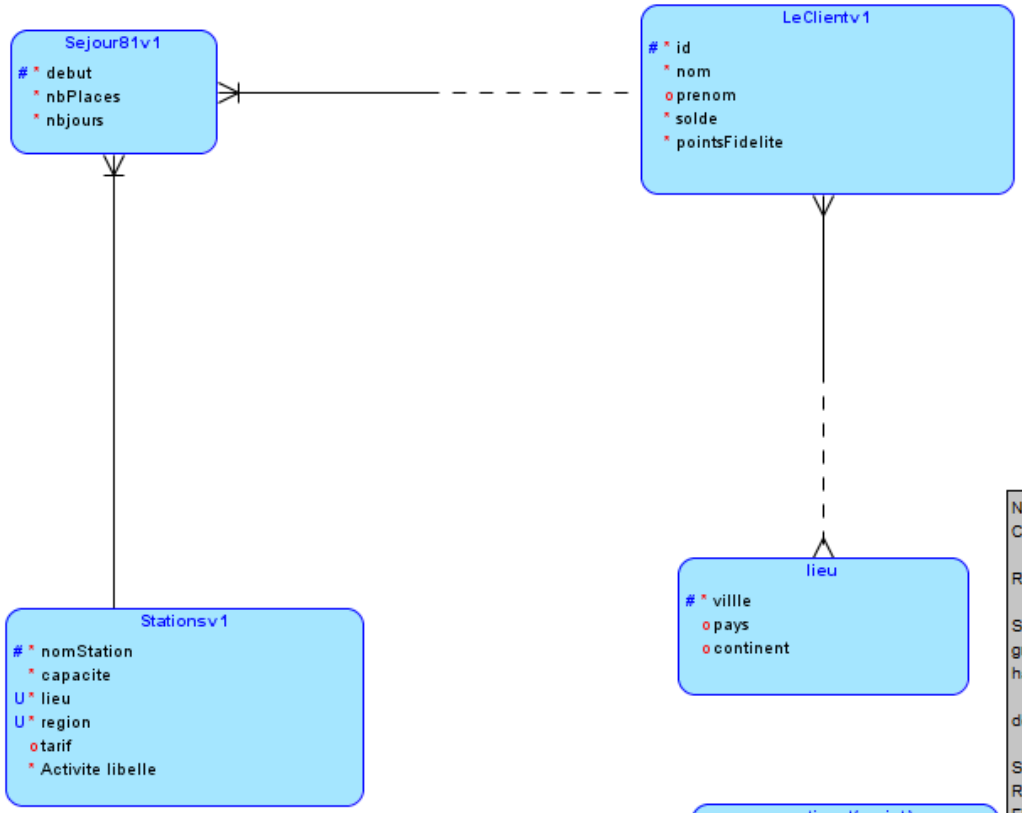

Sejour81
PF * idClient NUMBER (10)
PF * station VARCHAR2 (20)
P * debut DATE
* nbPlaces NUMBER (4)
* nbjours NUMBER (4)
Sejour81_PK (idClient, station, debut)
FK_LeClient (idClient)
FK_Station81 (station)
LeClient
P * id NUMBER (10)
* nom VARCHAR2 (20)
prenom VARCHAR2 (20)
* ville_region VARCHAR2 (40)
* solde NUMBER (10)
* pointsFidelite VARCHAR2 (6)
LeClient_PK (id)
Stations
P * nomStation VARCHAR2 (20)
* capacite NUMBER (10)
U * lieu VARCHAR2 (20)
U * region VARCHAR2 (20)
tarif NUMBER (10)
* Activite_libelle VARCHAR2 (20)
cle_station81 (nomStation)
cle_lieu_region81 (lieu, region)

Rappel

nbplaces >0
nbplaces<=50
nbjours>0
nbjours<=30

Recherche de sejours ne respectant pas ces contraintes plutôt raisonnables

```
select nbplaces from Séjour81 where nbPlaces<=0
select nbjours from Séjour81 where nbjours<=0
```



Rappel : toute transformation de schéma est en principe possible ; cependent, il est essentiel de la coupler avec un raisonnement hypothetique car l'interpretation du schéma pourrait être sensiblement differente de ce que les données suggèrent (si elles etaient corrects).

Les requetes reportées dans ce schéma offrent des exemples de raisonnement hypothètique.

D'autres requetes pourraient être utilisées si des nouvelles hypothèses sont formulées (par exemple, est ce que les types sont adaptés au contenu ? Est ce que lieu, region sont bien des données géographiques et pourrait on les asssocier à l'entité contenant les données géographiques des clients ? A quoi correspond le "tarif" ? Quelle est la veritable notion de séjour - car en effet la clé de sejour est fonction du client et de la date de debut ce qu'il implique que le sejour semblerait en effet le sejour d'un client specifique et que donc le sejour "vendu" serait en effet l'ensemble des sejours à la même station et les dates de debut serait les dates de départ possibles ? Est ce que le client est un passager -- ce serait utile car 2 passagers ne peuvent pas être dans 2 stations distinctes au même temps ?)

Les tranformations prennent leur fin lorque plus d'hypothèse n'est formulée.

capacité >= somme(nbplaces)

Recherche de sejours et stations ne respectant pas cette contrainte

```
select w.nomstation, z.dd,w.capacite

 from
  (select x.station, x.debut as dd, y.debut, x.nbplaces as nbp1, y.nbplaces as nb2
  from sejour81 x, sejour81 y
  where    (x.debut between y.debut and (y.debut+y.nbjours)) and
x.station=y.station or ((y.debut+y.nbjours) between x.debut and
(x.debut+x.nbjours))
  group by x.station, x.debut, y.debut, x.nbplaces, y.nbplaces) z, stations w
  where z.station=w.nomstation
  group by w.nomstation,z.dd, w.capacite

  having sum(z.nbp1)>w.capacite
```

--> comme l'on trouve 1 sejour et 1 station il faut decider
- si l'hypothèse (la contrainte) est maintenue et donc ces données seraient considerées erronées OU
- si ces données sont plausibles et donc l'hypothèse serait non validée et donc à supprimer ou à modifier (si modifiée, il faudra donc refaire le même raisonnement avec l'hypothèse modifiée)

Normalisation du Client
Client --> Ville_Region (DF)

Recherche de clients ne respectant pas la dépendence

```
Select count(*) from Client
group by id
having count(*)>1
```

décomposition de ville_region

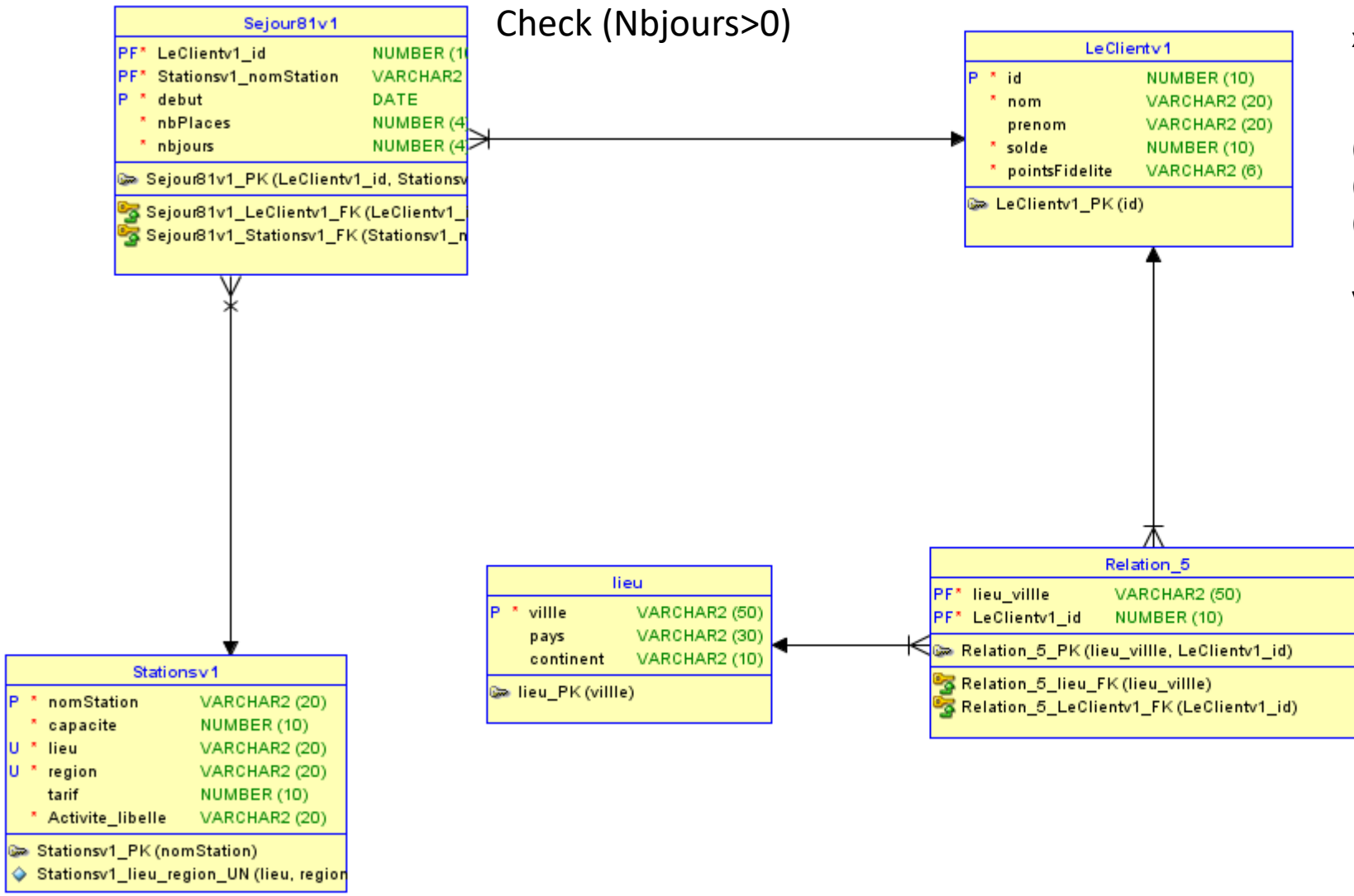
```
SELECT  REGEXP_SUBSTR(ville_region,'[^:]+',1,1),
REGEXP_SUBSTR(ville_region,'[^:]+',1,2)
FROM leclient
```

permettant de voir le contenu de "ville-region" et s'apércevoir qu'il y a bien une ville mais ensuite il ya un pays ou un continent.

Comment peut on distinguer entre continent et pays, une fois ces données extraites ?
Une possibilité simple est de disposer de certaines données independemment de la source.

Pour le "continent" cela est très simple car il suffit d'ecrire un script manuel permettant de lister les continents dans une table.

Rappel

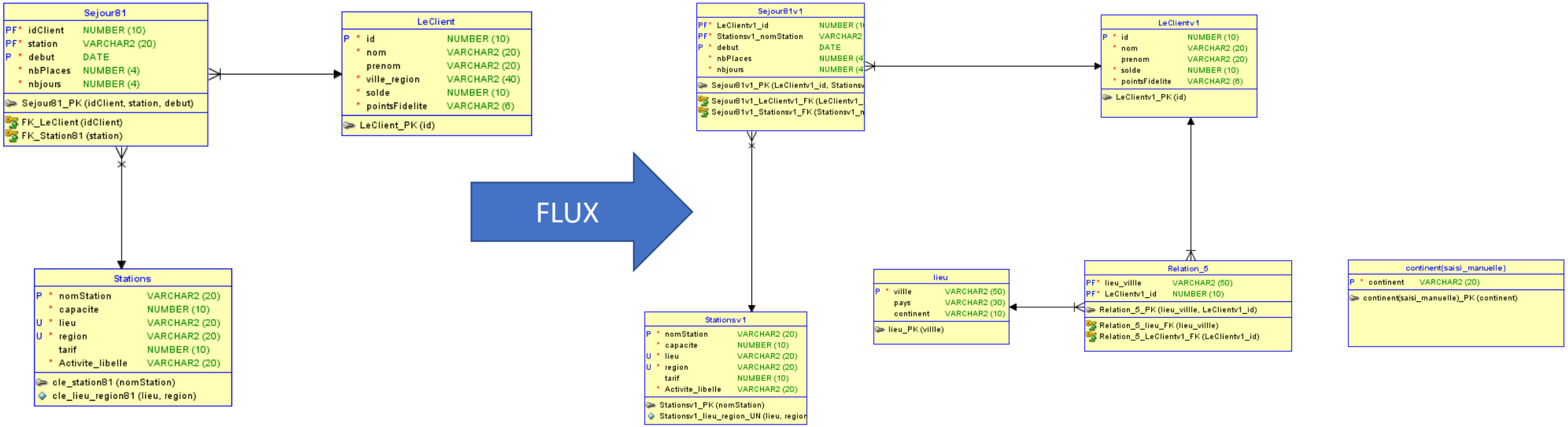


Create assertion… check (not exists (select w.nomstation, z.dd,w.capacite

from
(select x.station, x.debut as dd, y.debut, x.nbplaces as nbp1, y.nbplaces as nb2
from sejour81 x, sejour81 y
where (x.debut between y.debut and (y.debut+y.nbjours)) and x.station=y.station or ((y.debut+y.nbjours) between x.debut and (x.debut+x.nbjours))
group by x.station, x.debut, y.debut, x.nbplaces, y.nbplaces) z, stations w
where z.station=w.nomstation
group by w.nomstation,z.dd, w.capacite

having sum(z.nbp1)>w.capacite))

Flux d’extraction



Flux d’extraction

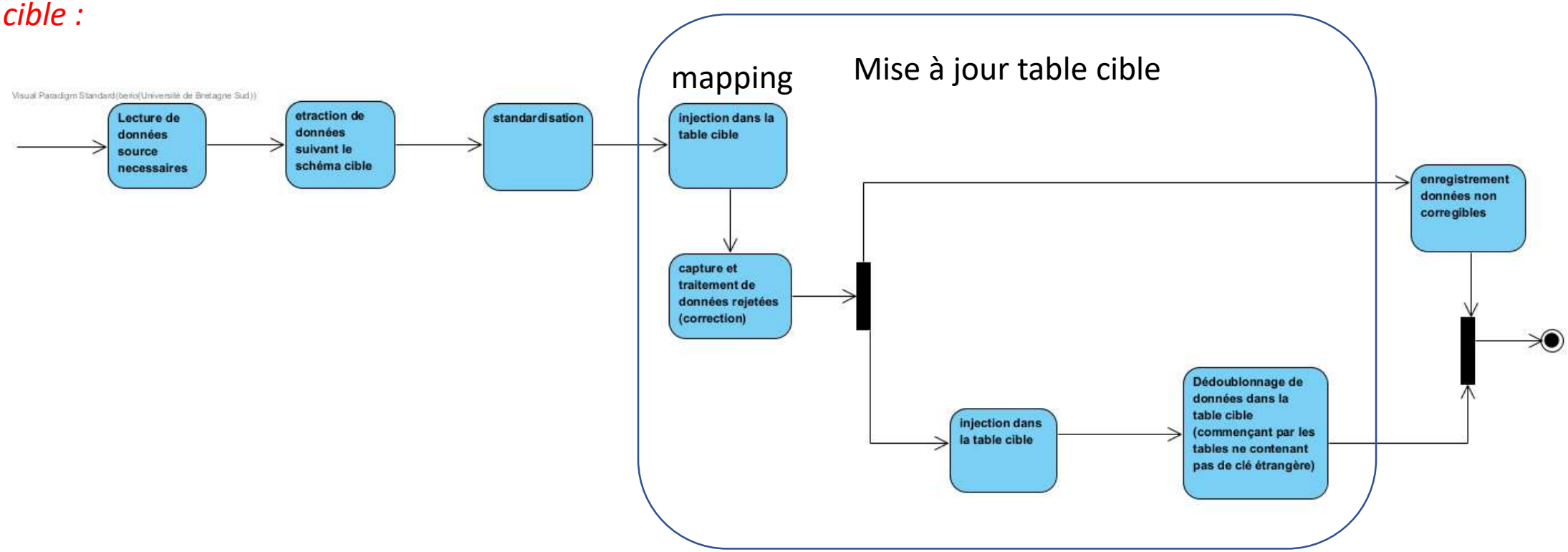
- Le flux d’extraction est un échange de données visant à créer de données cibles conforment à un schéma cible à partir de données sources
- Choix typiques :
  - Données extraites (cibles) stockées dans une BDD relationnelle
  - Données (potentiellement) incorrectes stockées dans fichier ou dans BDD relationnelle
  - Certaines contraintes sont directement associées aux tables relationnelles de la base contenant les données extraites
  - Pour d’autres contraintes, il se peut qui ne soit pas pertinent ou possible de les inclure dans la définition des tables de la base de données contenant les données extraites
- Problématiques à résoudre, typiquement après avoir finalisé la conception du flux :
  - Extractions successives
  - Changement de « clé » dans les sources (sauf si généralement non utilisé, excluant la reconfiguration des sources)

Mapping source-cible

- Il s’agit d’une spécification exécutable pouvant créer de données dans une table cible à partir de données sources (échange de données)
- Techniquement correspondant à une requête (type sql) et un insert (type sql)
- Un mapping ne prends pas forcement en compte des situations complexes, notamment le traitement de données erronées et le dédoublonnage
- En effet, toute situation complexe est traitée par une série de traitements de données (un flux de traitements)

Conception type d'un flux d'extraction (contraintes déclarées dans la table cible, hors problématiques)

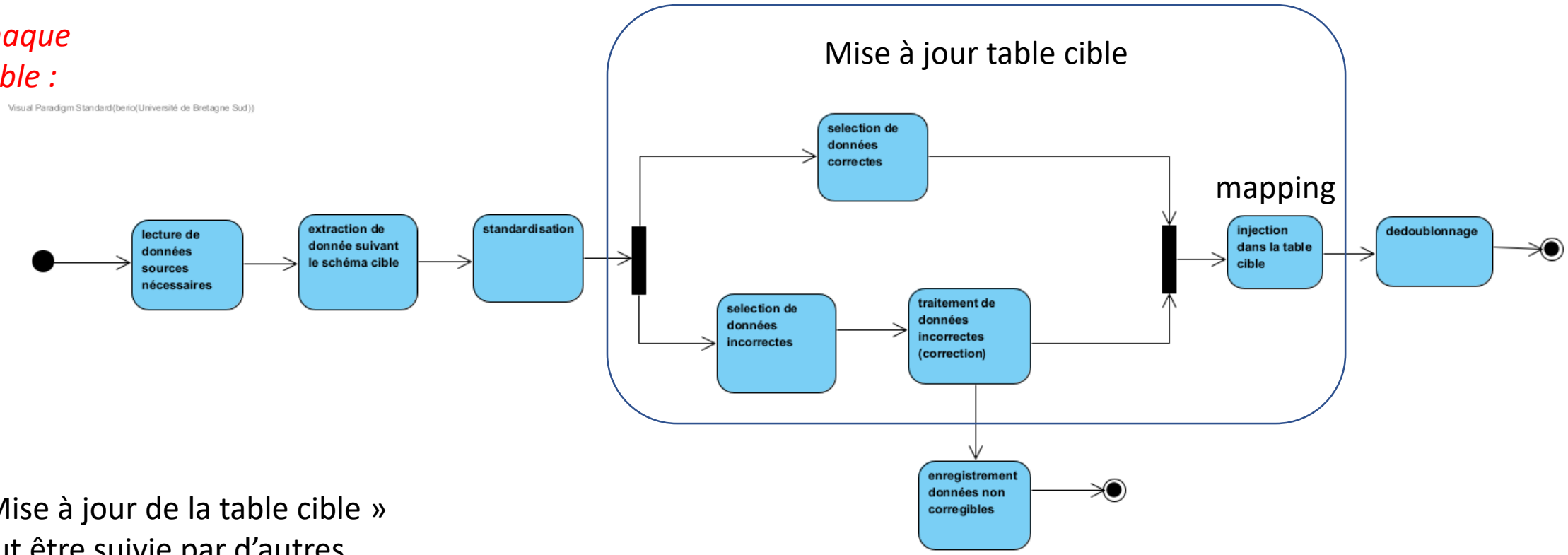
Pour chaque table cible :



« Mise à jour de la table cible » peut être suivie par d'autres activités avant dédoublonnage

Conception type d'un flux d'extraction (contraintes non déclarées dans la table cible, hors problématiques)

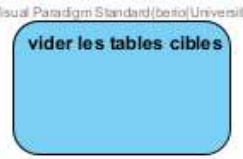
Pour chaque table cible :



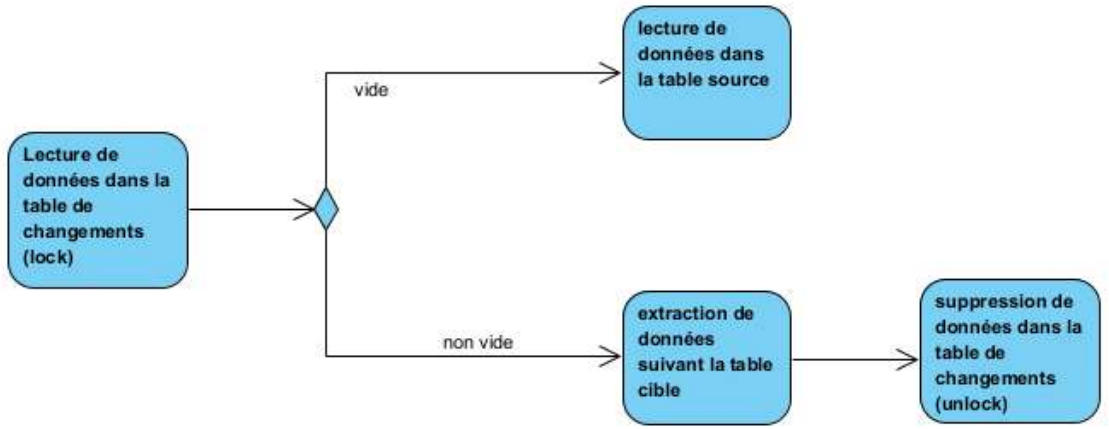
« Mise à jour de la table cible » peut être suivie par d'autres activités avant dédoublonnage

Extractions successives (source de type bdd)

1

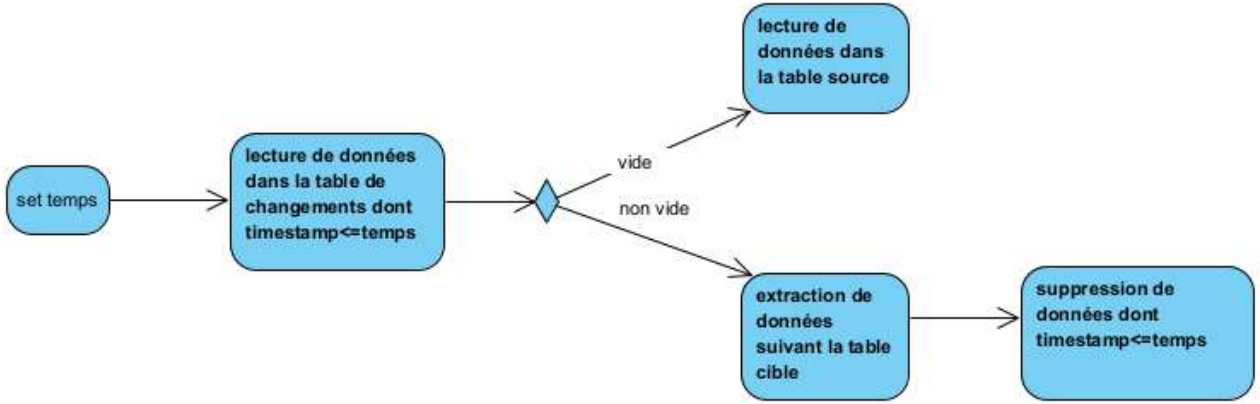


2



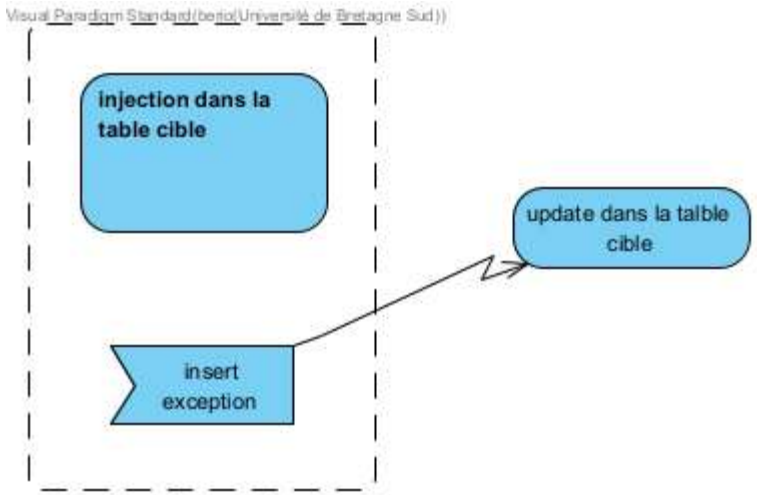
AVEC Contrôle de la concurrence

3



SANS Contrôle de la concurrence

Extractions successives (source de type bdd)



Détails des activités centrales (possibilités)

- Extraction de données suivant le schéma cible
  - Jointure
  - Split (ex. Regexp)
  - Concaténation (chaines de caractères)
  - Agrégation (numériques)
- Standardisation
  - Suppression de caractère séparateur, d'accent
  - Transformation tout majuscule/minuscule
  - Concaténation/Split
  - Enrichissement (introduction de nouvelles données)
  - Modification d'orthographe/terminologie
  - Traduction de langue
  - Changement/rajout d'unité de mesure
- Traitement de données incorrectes
  - Recherche de similarité avec de données supposées correctes (chaines de caractères, justement assez dépendent de ce que l'on considère être une chaine de caractères) et **substitution par les données correctes les plus similaires** – cela peut impliquer d'avoir préalablement traité de données d'une autre table
  - **Substitution de valeurs numériques aberrantes ou partielles** (d'autant que possible) par valeurs plausibles (moyenne, mode, somme,….)
- Dédoublonnage
  - Recherche de similarité de données dans la table
  - Fusion de données considérées doublons
  - Changement de clé

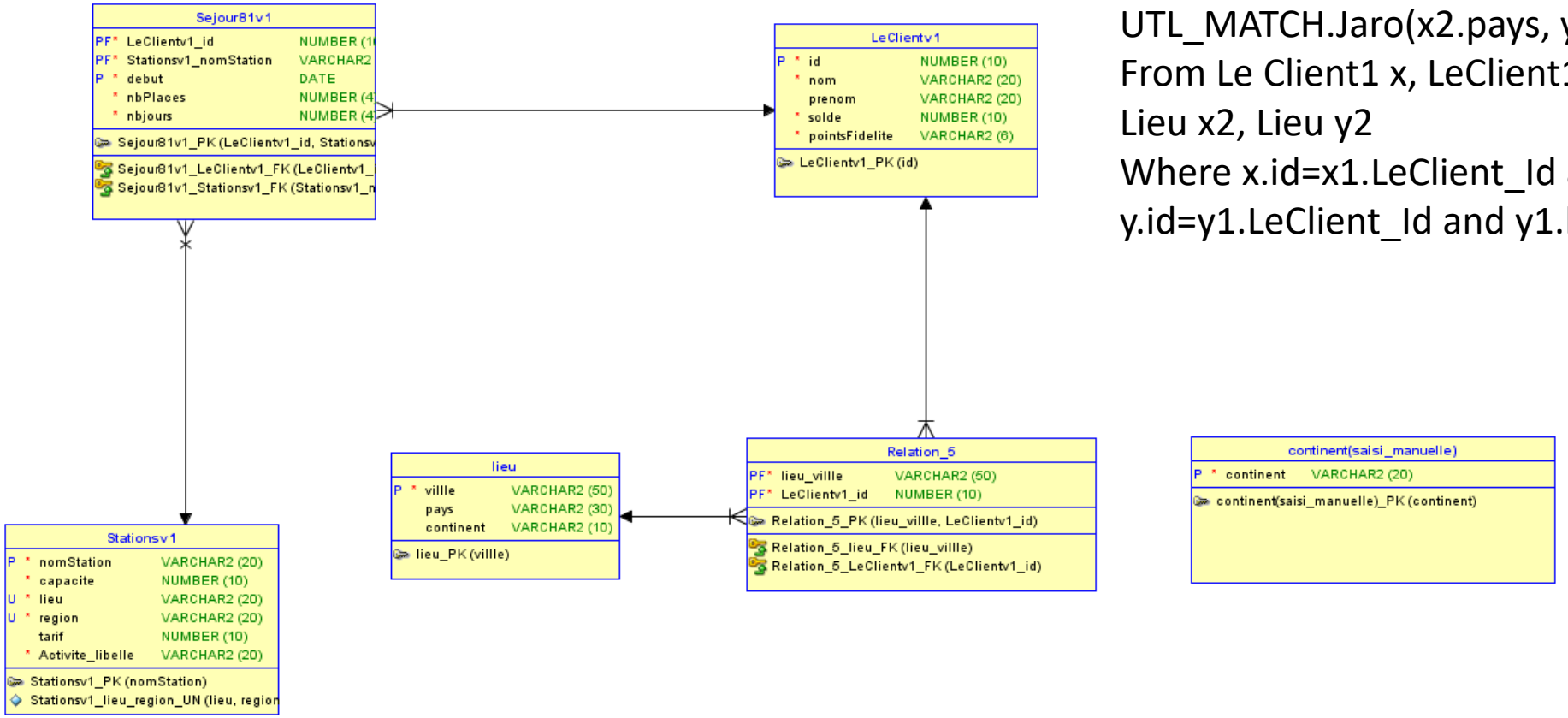
Focus : recherche de similarité

- Typiquement basée sur la similarité de chaines de caractères (un numérique peut être aussi considéré une chaine mais il peut ne pas être pertinent d'utiliser une similarité….les différences sont où peu significatives ou correspondantes à des erreurs ou conventions distinctes)
  - *LevDist(s1, s2)* = the minimum number of character insertions, deletions, and replacements necessary to transform *s*1 to *s*2.
  - *JaroSim* = $1/3\times(|\sigma|/|s1|+|\sigma|/|s2|+(|\sigma| - 0.5t)/|\sigma|)$, $|\sigma|$= number of matching characters, t = number of transpoditions required to get matching characters in the same position in both strings
  - *JaroWinklerSim(s1, s2) = JaroSim(s1, s2) +* $|\rho| \times f \times (1 - JaroSim(s1, s2)$
  (https://www.geeksforgeeks.org/jaro-and-jaro-winkler-similarity/)
  - Consinus similarity if strings are represented as numeric vectors
- Des combinaisons sont possibles (règle)
  - $\mathbf{0.2s_{nom}(x,y)}$ + ……+ $\mathbf{0.3s_{email}(x,y)}$ +….> 0,9
  - $\mathbf{0.3s_{nom}(x,y)}$ + $\mathbf{0.3s_{tel}(x,y)}$ + $\mathbf{0.1s_{ville}(x,y)}$ + $\mathbf{0.3s_{region}(x,y)}$….> 0,95
  - If $\mathbf{s_{email}(x,y)}$ >0,9 then match else if $\mathbf{s_{nom}(x,y)>0,9}$ **and** $\mathbf{s_{email}(x,y)}$ >0,7 then match else ……
- Et si vous disposez d'un sous-ensemble convenable de « training data » (un sous-ensemble de données pour lesquelles une réponse sûre existe doit de toute manière être disponible) une approche par apprentissage (supervisé) est aussi envisageable
- Si le « training data » n'est pas disponible,
  - il est possible de tenter l'algo de maximisation de la vraisemblance pour le déterminer
  - le « clustering » peut être envisagé (apprentissage non supervisé)
- Exploiter les relations entre données est aussi envisageable

Mise en œuvre d'une similarité

- Les différentes fonctions de similarité entre chaines de caractères peuvent être mises à disposition par l'ETL (par exemple réalisées en Java ou Python)
- Si elles ne sont pas disponibles, il faut les réimplanter (mais il suffit de rechercher le code en fonction du langage)
- Une alternative consiste à utiliser le SGBD dans utilisé pour la zone de staging pour appliquer la similarité
- Par exemple ORACLE, le package UTL_MATCH réalise un certain nombre de fonctions

Mise en œuvre d'une similarité (exemple d'ORACLE



Select x.ID, Y, ID, UTL_MATCH.Jaro(x.nom, y.nom), UTL_MATCH.edit_distance_similarity(x2.ville, y2.ville), UTL_MATCH.Jaro(x2.pays, y2.pays)
From Le Client1 x, LeClient1 y, Relation x1, Relation y1, Lieu x2, Lieu y2
Where x.id=x1.LeClient_Id and x1.lieu_ville=x2.ville and y.id=y1.LeClient_Id and y1.lieu_ville=y2.ville