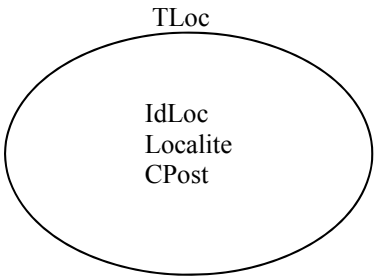
1.1. Elaboration de requêtes sur base d'une approche visuelle

1.1.1. Requêtes sur un ensemble unique.

Les requêtes sur un ensemble unique sont des projections et des sélections.



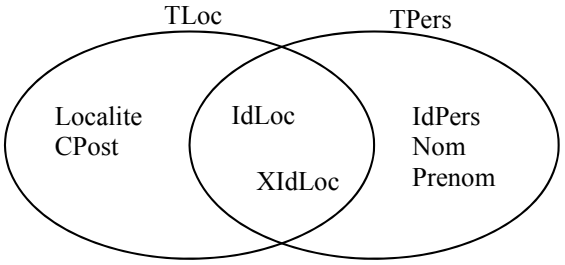

```
SELECT CPost, Localite FROM TLoc;
SELECT Localite FROM TLoc WHERE CPost >= '1300';
```

Les attributs souhaités pour la projection (clause **SELECT**) et ceux utilisés pour la sélection (clause **WHERE**) proviennent tous de l'ensemble unique spécifié dans la clause **FROM**. La démarche d'élaboration de telles requêtes se résume donc à choisir les attributs de la clause **SELECT** et ceux de la clause **WHERE**.

1.1.2. Requêtes sur une paire d'ensembles.

Les requêtes sur une paire d'ensembles sont des jointures réalisées sous forme de projections et sélections, ou par l'usage des clauses de jointures.

Lecture des personnes qui habitent une localité connue.




```
SELECT Nom, CPost, Localite
FROM TPers, TLoc
WHERE XIdLoc = IdLoc;
```

ou mieux :

```
SELECT Nom, Localite
FROM TPers
INNER JOIN TLoc
ON TPers.XIdLoc = TLoc.IdLoc;
```

Les attributs souhaités pour la projection (clause **SELECT**) sont choisis dans les ensembles parmi ceux qui n'appartiennent pas à l'intersection. Il n'est cependant pas interdit de choisir également des attributs dans l'intersection si leur extraction présente un intérêt pour l'application cliente de la requête. Si la requête est une sélection, la jointure est réalisée par la clause **WHERE** qui définit impérativement une comparaison entre les attributs contenus dans l'intersection. Il est possible d'affiner la sélection en complétant la clause **WHERE** par des conditions portant sur d'autres attributs des ensembles.

```
SELECT Nom, CPost, Localite
FROM TPers, TLoc
WHERE XIdLoc = IdLoc AND Localite LIKE  'Wa*';
```

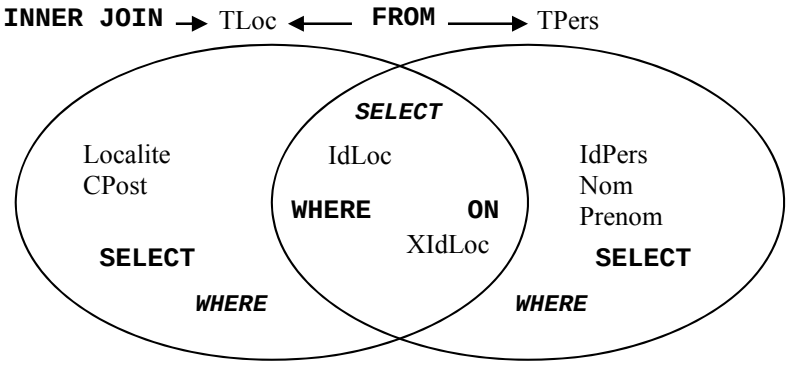
Si clause de jointure **JOIN** est employée, seul le nom de l'ensemble qui contient les attributs principaux est donné à la clause **FROM**. Le nom de l'autre est donné à la clause **JOIN** et la condition de jointure n'est plus définie dans une clause **WHERE**, mais bien dans la clause **ON** indissociable du **JOIN**. La clause **ON** doit contenir les noms des entités livrant les attributs de la condition. Il est possible d'affiner la sélection en ajoutant une clause **WHERE** vérifiant des conditions portant sur d'autres attributs des ensembles.

```
SELECT Nom, Localite
FROM TPers
INNER JOIN TLoc
ON TPers.XIdLoc = TLoc.IdLoc
WHERE Localite LIKE  'Wa*';
```

```
SELECT Nom, Localite
FROM TLoc
INNER JOIN TPers
ON TPers.XIdLoc = TLoc.IdLoc
WHERE Localite LIKE  'Wa*';
```

Il faut remarquer que les deux requêtes ci-dessus produisent exactement le même résultat. Il en résulte que le choix des ensembles pour les clauses **FROM** et **JOIN** est sans importance dans cet exemple. Mais ce n'est pas toujours le cas et il est pratique d'adopter un raisonnement qui soit approprié en toutes circonstances.
Le raisonnement qui prévaut ici consiste à déterminer si on souhaite les informations sur les personnes dont accessoirement leur localité, ou bien si on souhaite les informations des localités dont accessoirement leurs habitants. Dans le cas de la lecture des personnes avec leur localité, ce raisonnement conduit immanquablement à la requête présentée à gauche.

Il y a une relation entre les clauses SQL et les différentes parties des ensembles concernés par une requête.

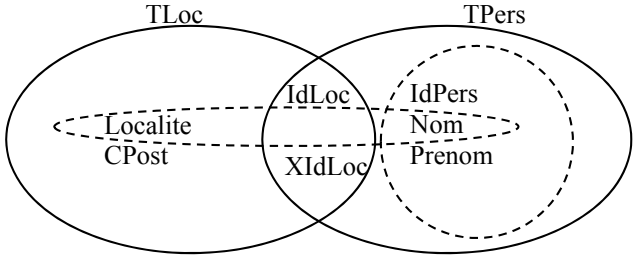


La jointure **INNER JOIN** livre uniquement les tuples qui satisfont à la condition de jointure exprimée dans la clause **WHERE** ou dans la clause **ON**. Les requêtes précédentes ne peuvent fournir des localités non habitées par les personnes de l'ensemble **TPers**, ni les personnes qui n'habitent pas une localité connue dans **TLoc**.

Il peut être nécessaire d'obtenir tous les tuples qui satisfont à la condition de jointure et aussi, distinctement, ceux d'un des ensembles ou des deux, pour lesquels la condition n'est pas réalisée.

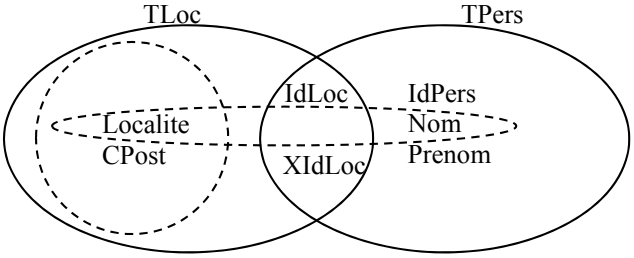
Lecture de toutes les personnes et de leur localité si elle est connue (lecture hiérarchique des parents).

```
SELECT Nom, Localite
FROM TPers
LEFT OUTER JOIN TLoc
ON TLoc.IdLoc = TPers.XIdLoc
ORDER BY Localite;
```



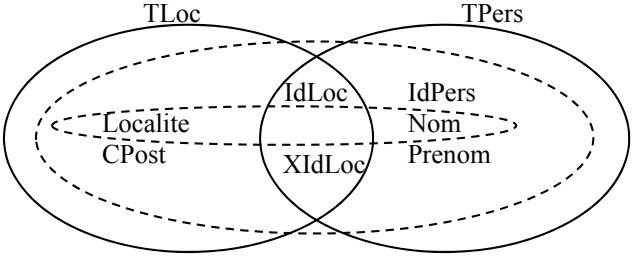
Lecture de toutes les localités et de leurs habitants connus (lecture hiérarchique des enfants).

```
SELECT Localite, Nom
FROM TLoc
LEFT OUTER JOIN TPers
ON TLoc.IdLoc = TPers.XIdLoc
ORDER BY Localite;
```



Lecture de toutes les personnes et de toutes les localités avec mises en évidence des correspondances.

```
SELECT Nom, Localite
FROM TLoc
FULL OUTER JOIN TPers
ON TLoc.IdLoc = TPers.XIdLoc
ORDER BY Localite;
```



ou, avec l'opérateur **UNION** :

```
SELECT Nom, Localite
FROM TPers
LEFT OUTER JOIN TLoc
ON TLoc.IdLoc = TPers.XIdLoc
ORDER BY Localite
UNION
SELECT Nom, Localite
FROM TLoc
LEFT OUTER JOIN TPers
ON TLoc.IdLoc = TPers.XIdLoc
ORDER BY Localite;
```

L'usage de la **FULL** jointure n'est pas supporté par tous les SGBD. La conversion d'une **FULL** jointure dans une forme mieux acceptée se réalise par le constat que les tuples souhaités sont ceux d'une **LEFT** jointure sur l'ensemble **TPers** associés à ceux d'une **LEFT** jointure sur l'ensemble **TLoc**. Il suffit donc d'écrire ces deux requêtes associées par un opérateur **UNION**. Il convient que les attributs soient spécifiés dans le même ordre dans chacune des clauses **SELECT**, faute de quoi il y aura de la redondance dans les tuples résultants.

La conception des **OUTER** jointures ne diffère pas de celle des **INNER** jointures. Les attributs souhaités sont spécifiés dans les clauses **SELECT** et choisis dans l'un des ensembles, ou dans les deux, et même si nécessaire dans l'intersection. La clause **FROM** désigne l'ensemble dont on souhaite extraire toutes les valeurs et la clause **LEFT OUTER JOIN** désigne l'autre ensemble. L'emploi de la clause **RIGHT OUTER JOIN** produit le même résultat que la clause **LEFT OUTER JOIN** si on inverse la désignation des ensembles entre les clauses **FROM** et **JOIN**. Les clauses **LEFT** et **RIGHT** désignent celui des ensembles qui doit livrer toutes ses valeurs : celui de gauche ou celui de droite par rapport à la clause elle-même à l'intérieur de la requête (la clause **FROM** étant toujours à gauche de la clause **JOIN**). Dans la mesure où on désigne l'ensemble principal dans le **FROM**, seule la clause **LEFT** reste nécessaire. La clause **ON** contient la condition de jointure qui est une comparaison d'attributs appartenant à l'intersection. Il est possible d'affiner la sélection en ajoutant une clause **WHERE** vérifiant des conditions portant sur d'autres attributs des ensembles.

```
SELECT Nom, Localite
FROM TPers
LEFT OUTER JOIN TLoc
ON TLoc.IdLoc = TPers.XIdLoc
WHERE Nom > 'B'
ORDER BY Localite
UNION
SELECT Nom, Localite
FROM TLoc
LEFT OUTER JOIN TPers
ON TLoc.IdLoc = TPers.XIdLoc
WHERE Localite LIKE 'Wa*'
ORDER BY Localite;
```