# Le sujet :

Dans une école, il y a des employées et étudiants. On demande de développer un système d’information, qui peut faire gestion des toutes personnes.

# Analyse des besoins :

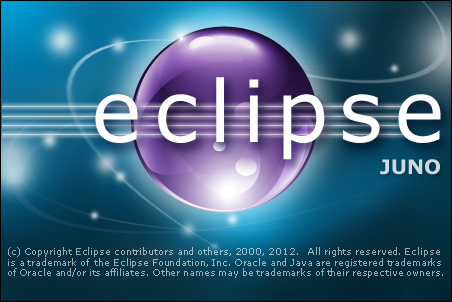
1. Les informations partagées : ID, nom, âge.
2. Les employées ont salaires, les étudiants ont notes.
3. On demande d’enregistrer toutes les informations dans les fichiers ou base de données, celles peuvent être ajoutées, modifiées, recherchées et supprimées.
4. Quand on recherche les informations, il peut l’affiche par ordre décroissant.
   1. L’employée : par salaire, l’étudiant : par notes.
   2. Si deux personnes ont les mêmes salaires ou notes, arrangement par âge décroissant.

# Préparation :

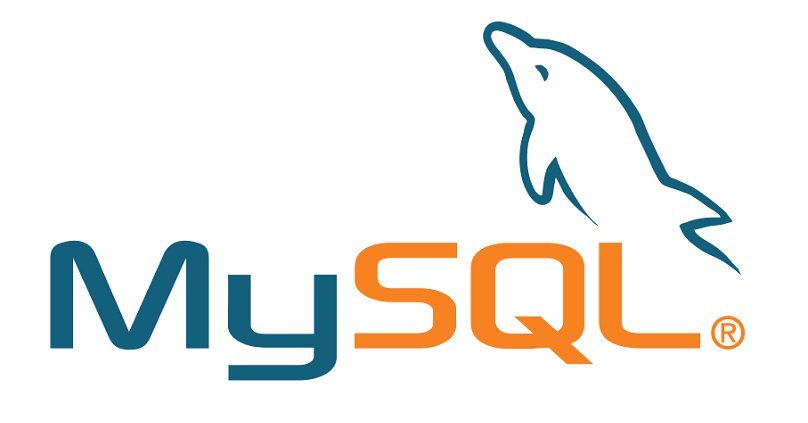
* Langage : Java, JDK1.7



* IDE : Eclipse



* Base de Donnée : MySQL



# Conception :

1. Définir une classe abstraite « Person », il y a des attributions : id, nom, âge. 2 classes « Worker » et « Student » qui héritent « Person ».

La classe doit réaliser l’interface « Comparable » et « Serializable », pour enregistrer les objets comparés dans une collection, et sauvegarder dans un fichier.



**public** **abstract** **class** Person **implements** Comparable, Serializable {

**private** String id;

**private** String name;

**private** **int** age;

**public** String getId () {

**return** id;

}

**public** **void** setId (String id) {

**this**.id = id;

}

**public** String getName () {

**return** name;

}

**public** **void** setName (String name) {

**this**.name = name;

}

**public** **int** getAge () {

**return** age;

}

**public** **void** setAge (**int** age) {

**this**.age = age;

}

}

1. Définir une interface DAO qui adapte import et export des fichiers et de la base. L’interface est implémentée par PersonDAOImplFile et PersonDAOImplBase.

Un proxy est défini, pour relier l’interface DAO et la classe spécifiée.



**public** **interface** PersonDAO {

**public** **boolean** doCreate (Person person) **throws** DAOException;

**public** **boolean** doUpdate (Person person) **throws** DAOException;

**public** **boolean** doDelete (Person person) **throws** DAOException;

**public** **boolean** doDeleteAll () **throws** DAOException;

**public** Set<Person> findAll () **throws** DAOException;

**public** Person findById (String id) **throws** DAOException;

**public** Set<Person> findByKey (String keyWord) **throws** DAOException;

}

1. Définir ID : Génération automatique. N + T + R

* N : 1 chiffre qui indique genre de personne : 1(employée), 2(étudiant).
* T : le temps : 20160101121314123(AAAAMMDDHHmmssxxx)
* R : 3 chiffres aléatoires : 258

Exemple : s20160101121314123258

**public** **class** Stamp {

**private** Calendar calendar = **null**;

**private** String type = **null**; // type.equals("1") ? Worker : Student

**public** Stamp(){

**this**.calendar = **new** GregorianCalendar();

}

**public** Stamp(String type){

**this**();

**this**.type = type;

}

/\*\*obtenir le temps actuel\*/

**public** String getTimeStamp(){}

/\*\*obtenir le temps avec 3 chiffre aléatoires\*/

**public** String getTimeStampRandom(){}

/\*\* ajouter 0 avant "str" si ce n'est pas suffi à "len" \*/

**private** String addZero(**int** str, **int** len){}

}

