# Introduction

Objectifs du cours

* Connaître les principales caractéristiques des bases de données
* Être capable d'effectuer une modélisation de données
* Comprendre la documentation spécifique de modélisation
* Maîtriser l'essentiel du langage SQL dit « standard »
* Savoir écrire une requête dans les règles de l’Art (standard, performances…)

## Une BDD dans un système d’information

### Définitions

* Le Système d’information : ensemble organisé de ressources (humaines, matérielles) et de procédures, permettant l'exécution des processus internes à une organisation via les manipulations de ses données.
* Le modèle relationnel : organisation consistant à stocker l'information dans des tables très précisément définies par leur schéma (leurs différentes colonnes, clés primaires, clés étrangères). Cela permet de ne pas stocker l'information plusieurs fois, et de pouvoir facilement consolider les données avec des requêtes SQL et des jointures.
* Base de données : Ensemble de données mémorisées sur des supports accessibles par un ordinateur pour satisfaire simultanément plusieurs utilisateurs de façon sélective et en temps très court. Elles constituent le cœur du système d’information.

La Base de données relationnelle correspond à la structure d’organisation des données correspondant au modèle relationnel.

* Donnée : un fait brut, impossible à interpréter tel quel. Exemple : un stock à 0
* Information : une donnée prenant du sens dans un contexte d'interprétation. Exemple : manque de carottes en stock
* Connaissance : une information analysée dans un contexte d'action. Exemple : un stock à reconstituer par l'achat de lots de carottes

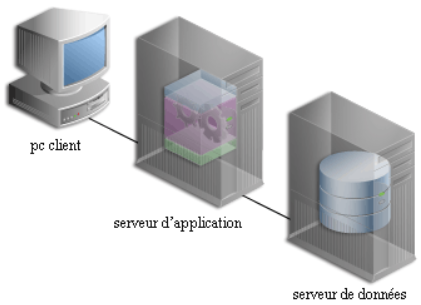
### Une base de données dans son environnement

Les bases de données sont fortement dépendantes du paysage informatique dans lequel elles baignent. Leurs objectifs divers et variés ont des impacts sur l’architecture technique et applicative, ainsi que sur les principes de leur modélisation.

Ces différents contextes amènent à penser la conception des bases de données de manière non dogmatique, en répondant au cas par cas (objectifs, contraintes, évolutivité, etc.).

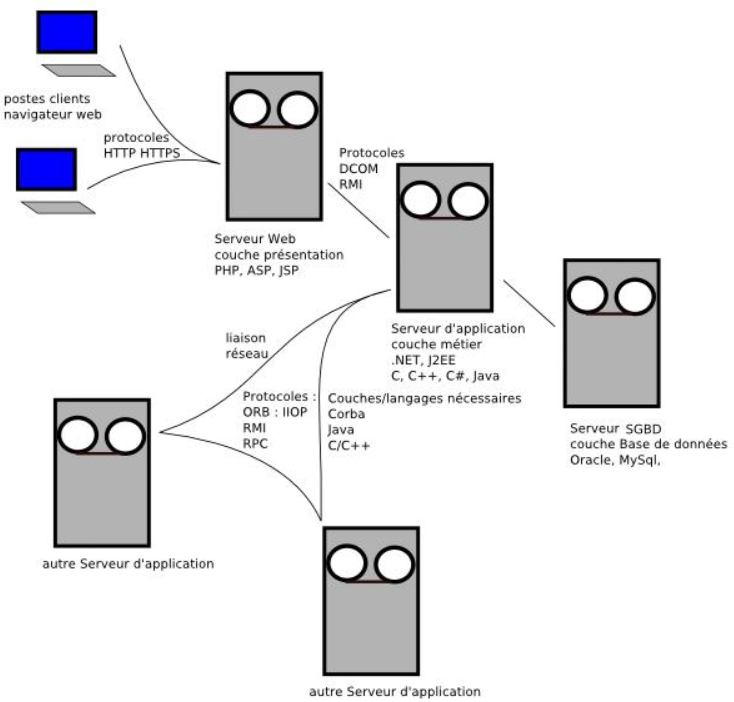
#### Architecture 3 tiers

Application du principe client/server dominant depuis les années 80. Le modèle relationnel est fortement adapté à cette structure.



#### Architecture n-tiers

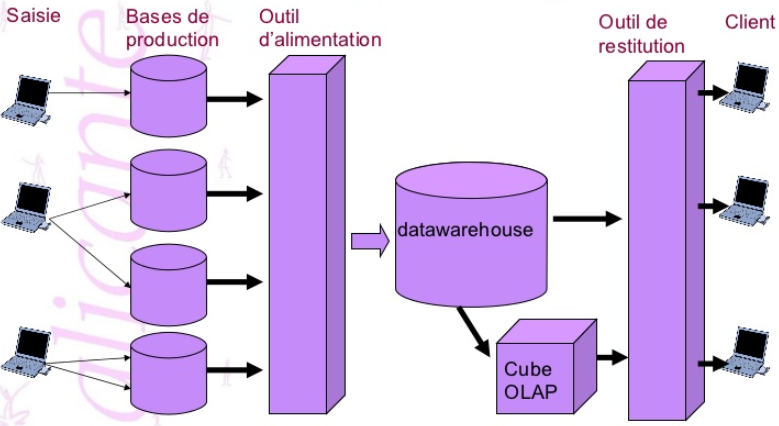
La performance des réseaux ont tendu à mutualiser les ressources, et donc à interconnecter les services applicatifs et informationnels. L’architecture n-tiers peut être vue comme une extension de l’architecture 3-tiers multi dimensionnelle.



#### Business intelligence

Métier de Développeur ETL (Extract – Transform – Load)

Le schéma relationnel est adapté dans un contexte transactionnel mais devient limité pour de l’analyse à grands volumes notamment pour l’aide à la prise de décision prospective à l’aide d’agrégation multiples et de visions selon les besoins. Cette nouvelle approche correspond à une forme de dénormalisation du modèle relationnel.



## Les normes SQL

Les bases de données de type relationnelles sont apparues dans les années 1980. Elles s’appuient sur les travaux développés par un chercheur, Mr Edgard Codd, travaillant chez IBM sur le modèle relationnel au début des années 1970. Les données sont organisées en tables distinctes sans niveau de hiérarchie. Il n’y a plus de pointeurs mais les données contenues dans les tables permettent de réaliser des liens entre les tables.

La tendance actuelle est aux systèmes qui utilisent plusieurs types de bases de données, comme SQL Server 2016. Il est ainsi possible d’écrire des requêtes qui portent sur des bases relationnelles et NoSQL.

Le langage SQL (Structured Query Language) signifie langage d’interrogation structuré. Il a été créé au début des années 1970 par IBM. C’est une start-up nommée Relational Software qui produira la première version commercialisable en 1979. Cette start-up est depuis devenue Oracle Corp.

Le langage SQL se décompose en plusieurs sous-ensembles :

* Le **DDL** pour Data Definition Language, qui regroupe les ordres utilisés pour créer, modifier ou supprimer les structures de la base (tables, index, vues, etc.). Il s’agit principalement des ordres CREATE, ALTER et DROP.
* Le **DML** pour Data Manipulation Language, qui regroupe les ordres utilisés pour manipuler les données contenues dans la base. Il s’agit principalement des ordres SELECT, INSERT, DELETE et UPDATE.
* Le **DCL** pour Data Control Language, qui regroupe les ordres utilisés pour gérer la sécurité des accès aux données. Il s’agit principalement des ordres GRANT et REVOKE.
* Le **TCL** pour Transaction Control Language, qui regroupe les ordres utilisés pour gérer la validation ou non des mises à jour effectuées sur la base. Il s’agit principalement des ordres COMMIT et ROLLBACK.

La première version de SQL normalisée par l’ANSI date de 1986.

Elle sera suivie de plusieurs versions plus ou moins importantes.

La norme SQL2 ou SQL92 est la plus importante. La majorité des SGBDR existants implémentent cette version.

Puis suivront plusieurs autres évolutions SQL-3, SQL:2003, SQL:2008 et SQL:2011 qui apportent chacune quelques fonctions complémentaires.

Une image contenant texte

Description générée automatiquement

Depuis quelques années, d’autres types de bases de données sont apparus avec les sites Internet utilisés par des millions d’utilisateurs :

* Base de données qui range les données en colonnes et non en lignes pour gérer des volumes de données énormes comme Bigtable, développée par Google, ou Cassandra, utilisée par Facebook.
* Base de données NoSQL, comme Redis, plus souple avec un schéma défini de façon imprécise, qui se préoccupe de la cohérence finale.
* Base de données de documents, similaire au type de base NoSQL. La plus connue et utilisée est MongoDB.

Chaque fournisseur de SGBDR a implémenté à sa façon le langage SQL et a ajouté ses propres extensions. Attention à la compatibilité !

Le site db-engines.com actualise régulièrement les informations sur la popularité des bases de données, globalement et par catégorie. Il explique sa méthodologie de classement.

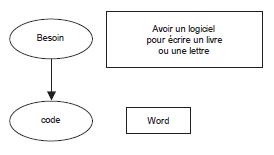
Une image contenant texte

Description générée automatiquement

# Modélisation des données

## Cycle de vie d’un projet

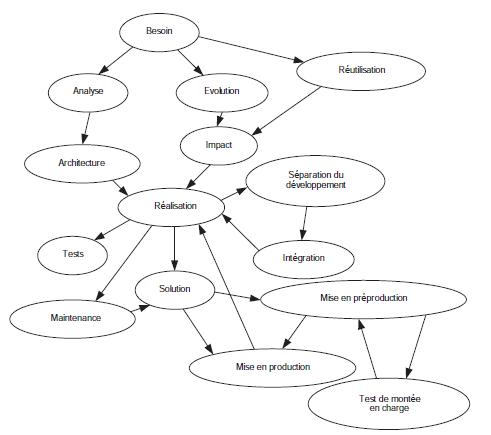
### Le début de la réflexion : Besoin -> réalisation…



### …et les problèmes surviennent

* Comment identifier correctement le besoin -> Analyse du besoin
* Le besoin s’inscrit-il dans la durée, est-il évolutif, donc impact sur le code -> Réutilisation et évolutivité du code
* Paysage applicatif et architectural de l’application -> impacts en entrée et sortie de l’application
* Le code est-il correct dans tous les cas de figure -> Tests du code (unitaires, intégration, qualité, performance)
* Les utilisateurs vont-ils modifier leurs processus avec l’application -> suivi des besoins, intégrer les évolutions et améliorations
* Comment va être géré le workflow des remontées utilisateurs (bugs) -> Processus de maintenance
* Formation utilisateur -> Documentation à différents niveaux
* Comment les utilisateurs vont-ils pouvoir piloter l’application -> Documentation d’installation de l’application et des processus courants (paramétrages)
* Quelle est la politique de mise en production et de correctifs -> directe, par pallier, patchs, montée de versions…

Ce qui nous donne au final, une fois considéré toutes ces activités lors de la durée de vie du projet, plutôt ce type de schéma :



### Solution projet : Le cycle de l’abstraction

* Etudier et comprendre l’Univers réel
* Le modéliser pour le représenter simplement
* Concevoir la solution informatique
* Implémenter la solution

En méthodologie classique, ce cycle de l’abstraction est représenté par 3 étapes majeures :

|  |  |  |
| --- | --- | --- |
| Niveaux | Données | Rôle |
| Conceptuel | Modèle Conceptuel des Données | Analyse/Conception |
| Logique | Modèle Logique des Données | Traduction en logique informatique |
| Physique | Modèle Physique des Données | Réalisation |

## Le modèle conceptuel des données = MCD

Le MCD introduit la notion d’entités, de relations et de propriétés. Il décrit de façon formelle les données utilisées par le système d’information. La représentation graphique, simple et accessible, permet à un non informaticien de participer à son élaboration. Les éléments de base constituant un modèle des données sont :

* Les propriétés sont reliées par des ID (clés primaires)
* Les entités (ou objets)
* Les relations entre les entités ou associations

Cycle de création d’un objet dans une BDD :

* CRUD = Create Read Update Delete

***C = create***

Insert into clients values (1,2,3,4)

***R= read***

Select

From

Where

***U = Update***

Insert into clients values(‘Elek’) Where num\_client=CL02

D= Delete

Delete from clients where name=’Elek’

### Les propriétés

Les propriétés sont les informations de base du système d’information.

Un client possède un numéro de client, un nom, un prénom, habite à une adresse précise, etc. Ces informations élémentaires essentielles sont des propriétés.

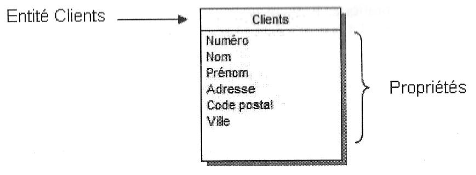
Les propriétés disposent d’un type. Elles peuvent être numériques, représenter une date, leur longueur peut être aussi définie. Par exemple : le nom est une propriété de type alphabétique et de longueur 50, c'est-à-dire que la valeur saisie ne comportera aucun chiffre et ne dépassera pas 50 caractères. Les propriétés doivent être atomiques (insécables), relativement au système considéré et non redondante.

Les types ne sont pas décrits au niveau conceptuel, car ce niveau est trop proche de la définition du système physique.

Les propriétés sont des données brutes et non des calculs.

### Les entités (ou objets)

L’entité est l’objet de gestion d’un nombre de propriétés identifiées, portées par celle-ci. Comme il est aisé de le constater, les clients sont définis par certaines propriétés (numéro, nom, prénom…). Le fait de les regrouper amène naturellement à créer une entité Clients. Le symbolisme retenu est le suivant :



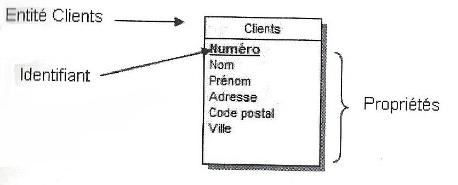
L’une des propriétés a un rôle précis, c’est l’identifiant nommé aussi la clé primaire.

L’identifiant permet de connaitre de façon certaine et unique l’ensemble des propriétés qui participent à l’entité. Par exemple, le fait de connaitre la ville d’un client permet-il de connaitre son nom ? La connaissance du nom permet-il de connaitre la ville ? Non.

Il faut donc trouver, ou inventer, une propriété qui lorsque sa valeur est connue permet la reconnaissance de l’ensemble des valeurs qui s’y rattachent de façon formelle.

Ainsi, lorsque le numéro du client est connu, son nom, son prénom et toutes les valeurs des autres propriétés qui s’y rattachent sont connues de façon sûre et unique.

Au niveau du formalisme, cette propriété se souligne. Voici le schéma modifié de l’entité Clients :



### Les relations

#### Définition

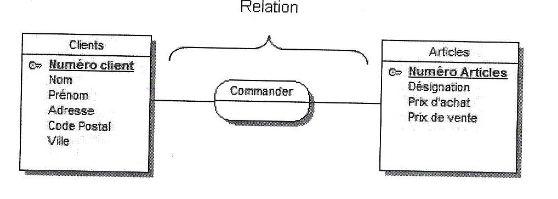
Les entités regroupent un ensemble d’informations élémentaires. Les entités sont liées entre-elles.

Une entité ‘faible’ représente le lien entre entités ‘fortes’. Elles sont la relation entre elles.

Par exemple :

Un client peut commander des articles.

Si nous analysons cette phrase, on distingue 2 entités Clients et Articles) et un verbe (commander) qui indique une action les reliant. Formalisons cette phrase avec Merise.



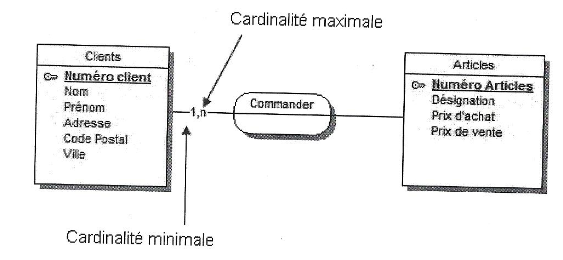
#### Les cardinalités

En reprenant la même phrase, on peut y distinguer une information supplémentaire : la cardinalité. **Un** client peut commander **des** articles.

Les cardinalités expriment le nombre de fois où l’occurrence d’une entité participe aux occurrences de la relation.

* Combien de fois au minimum un client peut-il commander un article ?
* Combien de fois au maximum un client peut-il commander un article ?

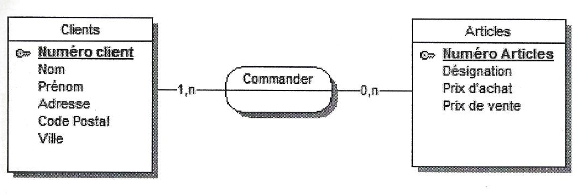
Voilà comment formaliser la réponse à ces 2 questions, selon le contexte :



De même, en considérant l’article :

* Combien de fois au minimum un article peut-il être commandé par un client ?
* Combien de fois au maximum un article peut-il être commandé par un client ?

La réponse à ces 2 questions selon le contexte :



La cardinalité minimale (0 ou 1) exprime le nombre de fois minimum qu’une occurrence d’une entité participe aux occurrences d’une relation.

La cardinalité maximale (1 ou n) exprime le nombre de fois maximal qu’une occurrence d’une entité participe aux occurrences de la relation. Si le maximum est connu, la valeur doit être inscrite au niveau du MCD.

A ce stade, il est encore souvent question d’interprétation. Ces interprétations traduisent la plupart du temps des choix fonctionnels structurants que l’équipe de développement pourra ainsi anticiper.