INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- **Présentation globale des bases de données**
  - présentation, architecture, définitions, objectifs, historique, références.
- **Modélisations des données**
  - Systèmes hiérarchiques, réseaux, relationnels
- **Le modèle relationnel**
  - concepts, dépendances fonctionnelles, règles d 'intégrité, formes normales
- **Le langage SQL (Structured Query Language)**
  - définition de données, manipulation de données, contrôle des données, les vues.
- **Un aperçu de « NoSQL ».**
  - Définition des « index », manipulation de données.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- Cours une idée générale
- TD un approfondissement
- TP une mise en application notée

- Pas d'examen final
- 2 à 3 évaluations,
- des tests « surprises » de compréhension sous forme de QCM et d'autres formes si le distanciel ou l'hybride devait revenir.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

I Présentation globale des bases de données

Qu'est-ce qu'une « Base de Données (BD) » ?

- Une **base de données (DataBase)** est une collection de données structurées sur des entités (objets, individus) et des relations dans un contexte (application) particulier.
- Autre définition : un ensemble structuré de données apparentées qui modélisent un univers réel ou abstrait.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

« Base de Données» Késako ?

- Une BD est faite pour enregistrer des faits, des opérations au sein d 'un système d'information.
- SGBD : un système de gestion de base de données (français) ou DBMS : DataBase Management System est un ensemble de logiciels qui facilite la création et l'utilisation de bases de données.
- Les données sont définies, administrées et gérées en utilisant des langages fondés sur des modèles de données.
- La base de donnée est un élément central dans le SI d'une entreprise (**Système d'Information**). Elle permet d'obtenir les éléments nécessaires à la prise de décision.

Les bases de données : une histoire de plus d'un demi siècle

- 1969 Première Standardisation du modèle réseau **CODASYL**
  (Conférence On Data Systems Languages)

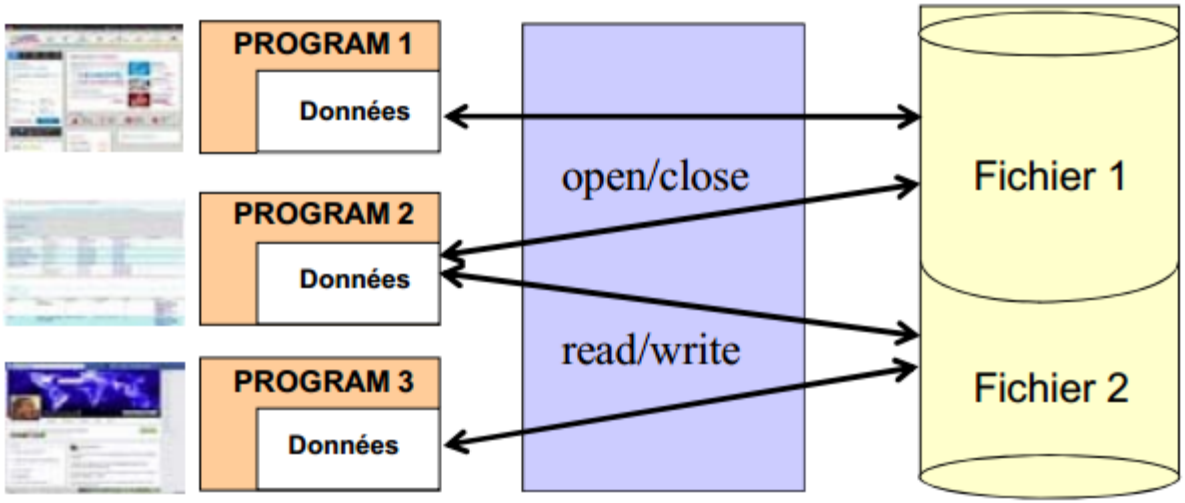
- 1970 : **TED CODD** définit le modèle relationnel pour IBM ce qui induit deux projets de recherches majeurs :
  - **INGRES** (Berkeley Calfornia) qui devient **POSTGRES** (logiciel libre!) qui devient ILLUSTRA racheté par **INFORMIX**.
  - System R pour IDM qui devient **DB2** qui inspire **ORACLE** (Unix).

http://scihi.org/codd-relational-database-model/

Différentes Approches historiques…

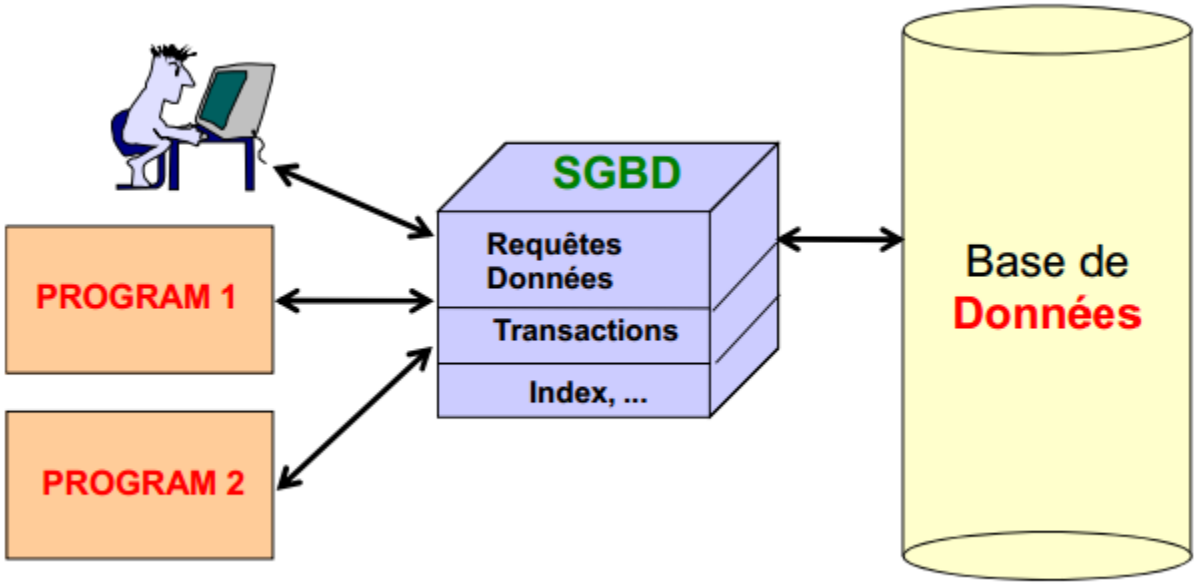
Approche Fichier

- 4 opérations simples ouvrir/fermer et lire/écrire
- Des méthodes d'accès (indexées, hachées, directes, …)
- utilisation simultanée par plusieurs programmes => difficile !
- La gestion de données structurées dans des fichiers représente un coût important de développement.



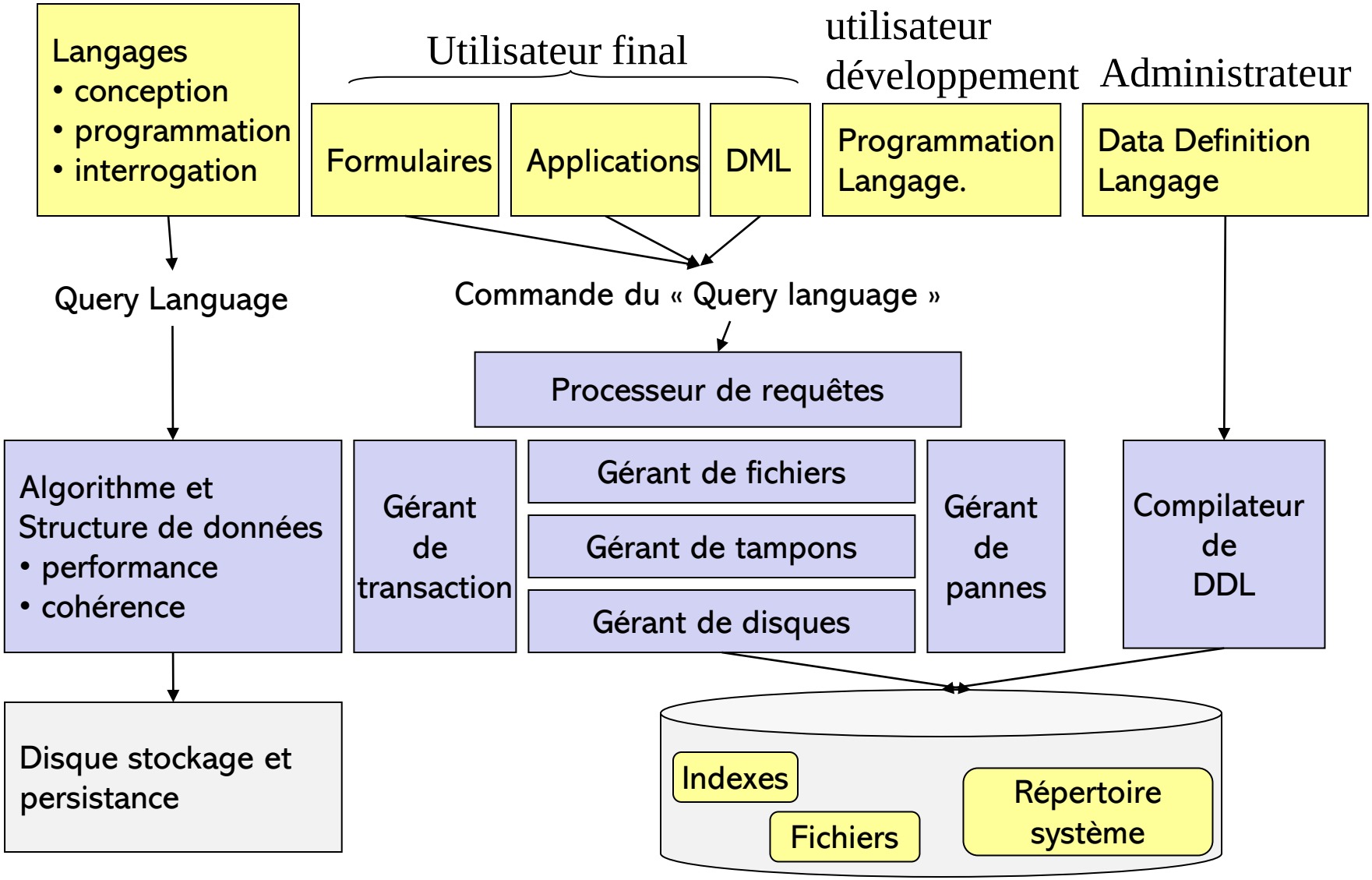
Différentes Approches historiques…

Approche SGBD



- structuration des données à travers un schéma de données
- Indépendance entre les programmes et la gestion des données
- Données partagées
- contrôle de cohérence (logique/physique à travers les schémas et les transactions).
- Performance d'accès élevée

ARCHITECTURE D'UN SGBD (DBS) EN 3 NIVEAUX

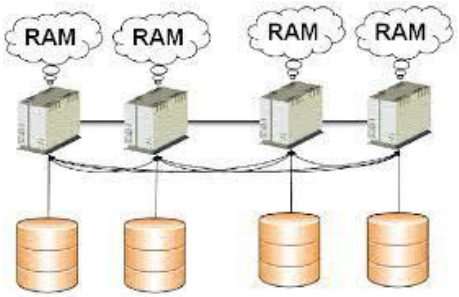


Les outils des SGBD

- **Décrire : Langages de définition de données (LDD ou DDL)**
  - conception haut niveau; aperçu des données et des traitements.
  - pour définir les schémas externes (vues), logiques et physiques
- **Manipuler : Langage de manipulation de données (LMD ou DML)**
  - langage déclaratif pour interroger (requêtes ou request) et mettre à jour :
    - dire QUOI sans dire COMMENT.
    - exemple : Quels sont les noms des produits ayant un prix < 100€.
  - Langage Autonome (ex: SQL) ou Intégré dans un langage de programmation (ex: dans une API JDBC)
- **Contrôler les données : Langage de contrôle des données (LCD ou DCL)**
  - exemple : le prix doit être compris entre 1 et 100€ lors de l'insertion dans la base.

Les outils de gestion des SGBD

- **Partage :**
  - contrôle des accès concurrents des transactions à la base
  - assure la cohérence de la base.
- **Sécurité**
  - reprise sur panne
  - gère un journal des évènements
- **Performance d'accès**
  - index (hachages, arbres balancés,...)
  - clusterisation



- **Indépendance logique**
  - vues différentes des mêmes données pour des application différentes
  - modification des schémas logiques sans impact sur les applications .
- **Indépendance physique**
  - modification des structure de stockage sans impact sur les applications.

Définitions des langages et interface SGBD

- **Langages de conception (modèle Entité-Association, UML)** : utilisation pour la conception « haut niveau » aperçu des données et des traitements.
- **Langage de la base de données (SQL, algèbre relationnel, N.O. SQL…)**
  - langages déclaratifs : spécifier QUOI et non comment.
  - Utilisation pour la définition de schémas, d 'interrogation, de mise à jour et d 'administration.
- **Langage de programmation comme JAVA, JSP, PHP, ….**
  - Langages avec une interface SQL, exemple JDBC.
  - Utilisation pour la programmation d'applications avec accès aux Données.
- **Langage de programmation de bas niveau : C, C++,Visual Basic,**
  - mise en œuvre d'un accès au SGBD via une bibliothèque implémentant les opérateurs physiques, spécifiques de la base.

- **1976 : Peter Chen** (UCLA/ IBM) définit le modèle Entité-Association (E/A) ou Entity Relationship (E/R)
- **1980** maturation des technologies relationnelles :
  - Standardisation des langages : **SQL (Structured Query Language)**.

- **1990**
  - technologie relationnelle avec support de distribution et parallélisme
  - Création « **ODMG** » Object Database Management Group (1991)
  - Emergence de la modélisation **UML** Unified Moddeling Language (1995).
  - Emergence des outils Neo.

GraphQL + neo4j

- Fin 1990
  - Émergence de nouveaux domaines : **entrepôts de données**, décisionnel (**Business Object. - TerraData**), « **www** » , multimédias, mobiles, etc…

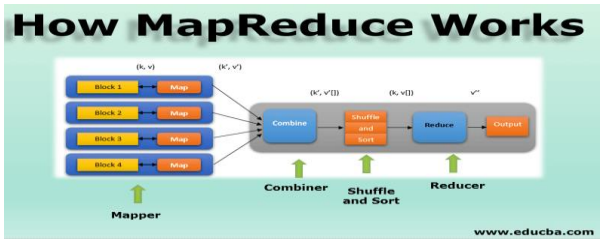
  - Émergence de nouveaux domaines :
    - entrepôts de données,
    - décisionnel (SAP Business Object - TeraData),

- Besoins autour du web : Moteurs de recherche, multimédias, mobiles,…

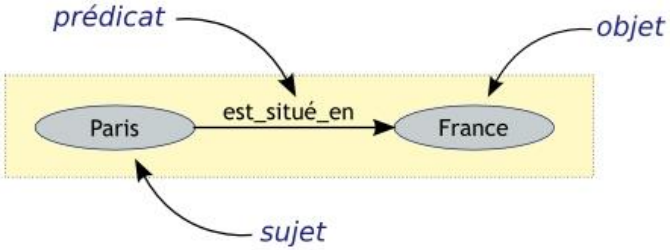
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

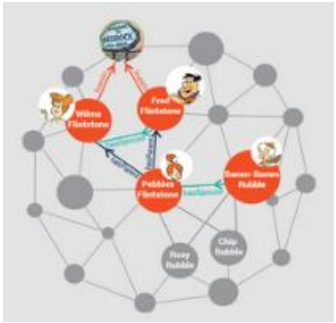
- **Années 2000 et 2010**
  - Apparition de **XML** et de nouvelles architectures (ex : P2P)

  - **NoSQL: Not Only SQL**
    - hbase,
    - MongoDB,
    - Kibana hadoop
    - **MapReduce** (Google),



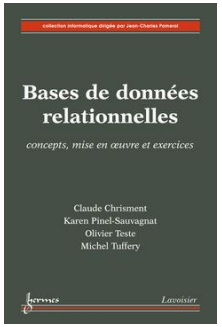
    - **RDF** (Ressource Data Framework : web sémantique), ….





Initiations aux bases de données -16- Références

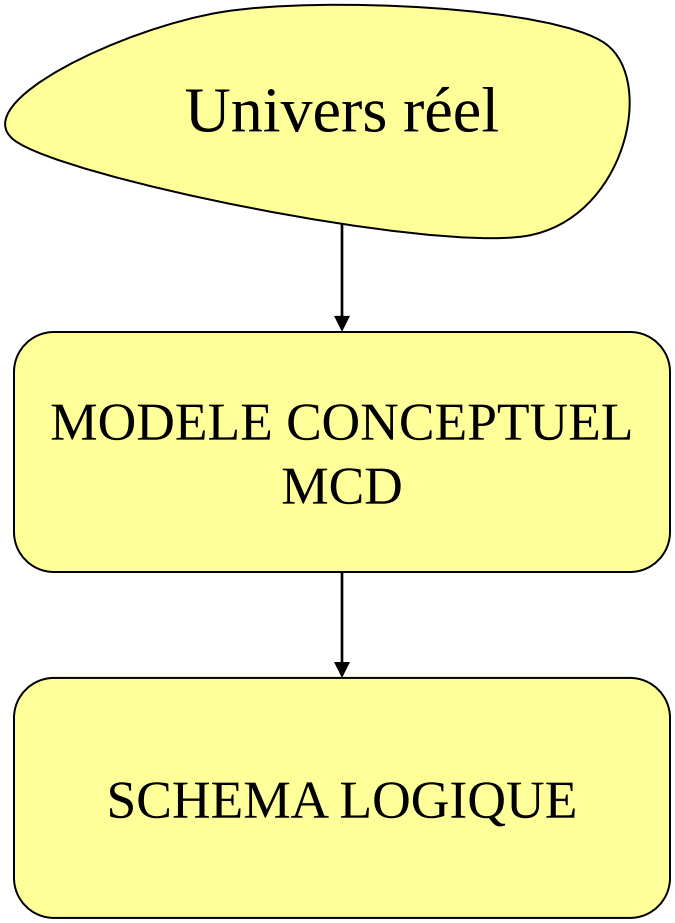
- . Chrisment et.al.,
  - Bases de données relationnelles,
  - Ed. Hermes - Lavoisier
    Données (SGBD)
- G. Gardarin.
  - Bases de données - objet et relationnel.
  - Ed. Eyrolles.
- J.L. Hainaut,
  - Bases de données : concepts, utilisation et développement,
  - Ed. Dunod
- S. Abiteboul, R. Hull, V. Vianu,
  - Les fondements des bases de données,
  - Ed.Vuibert



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

II Présentation globale des bases de données

Initiations aux bases de données -18- MODELES



Modèles sémantiques
- Modèles orientés « conception »
- Entité-Association,
- Merise
- UML….

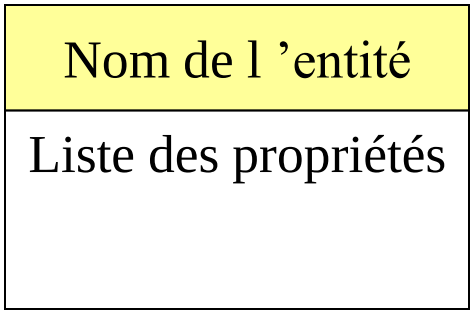
Modèles de BD
- Hiérarchique,
- Réseau
- Relationnel …

Les modèles de BD sont souvent trop limités pour représenter directement le monde réel.

- **Conceptuel : le modèle entité Association E/A (E/R)**

Il s ’agit du formalisme graphique retenu par l'ISO pour décrire l'aspect conceptuel des données à l’aide d’entités et d’associations sous forme de schémas conceptuels.

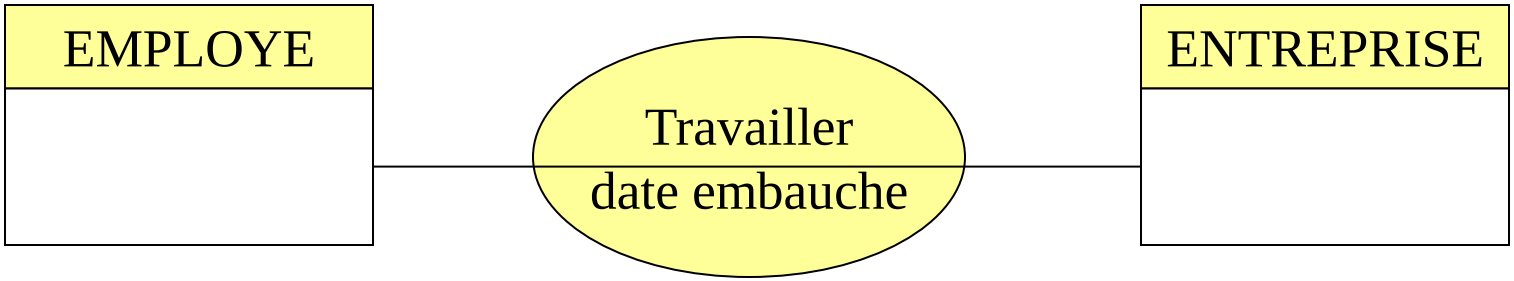
– **Entité** :représentation d ’objet matériel ou immatériel, exemple : un employé, un projet, un bulletin de paie.



– **Type d ’entité** : correspond à un regroupement d’entités
  - exemple : les développeurs sont des instances du type d ’entité générique EMPLOYE

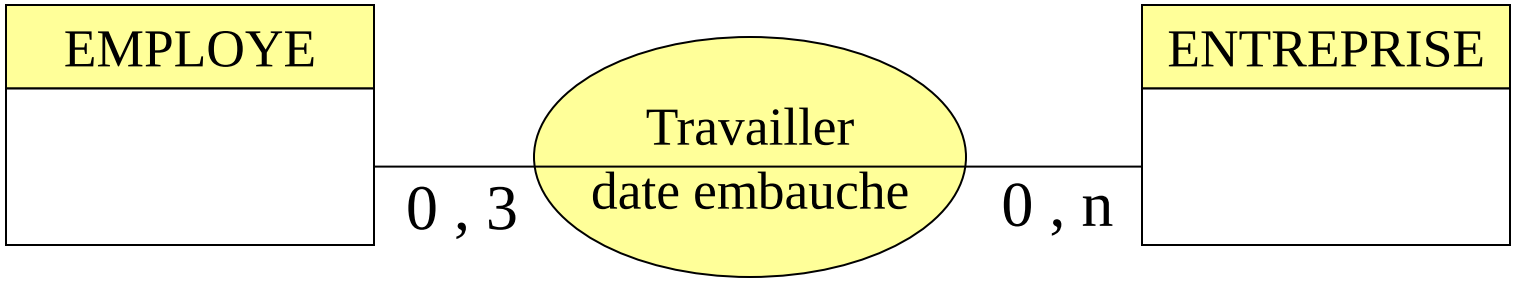
– Les **propriétés** sont les données élémentaires relatives à une entité, par exemple : nu numéro d ’employé, un nom, un prénom,etc. Elles sont appelées **attributs** ou caractéristiques de cette entité.

- **Conceptuel : le modèle entité Association E/A (E/R)**
  - l'**identifiant**
    - propriété ou groupe de propriétés qui sert à identifier une occurrence de cette entité de façon unique.
  - Les **associations**
    - représente le lien entre les entités. Elle peuvent avoir des propriétés !



  - ce modèle a pour objectif :
    - la détection d 'erreur de conception avant développement
    - la traduction automatique dans un modèle logique
  - Il est supporté par les outils de type CASE.
  - *Il correspond à la partie modélisation de données dans UML.*

- **Conceptuel : le modèle entité Association E/A (E/R)**
  - *Les **cardinalités**:*

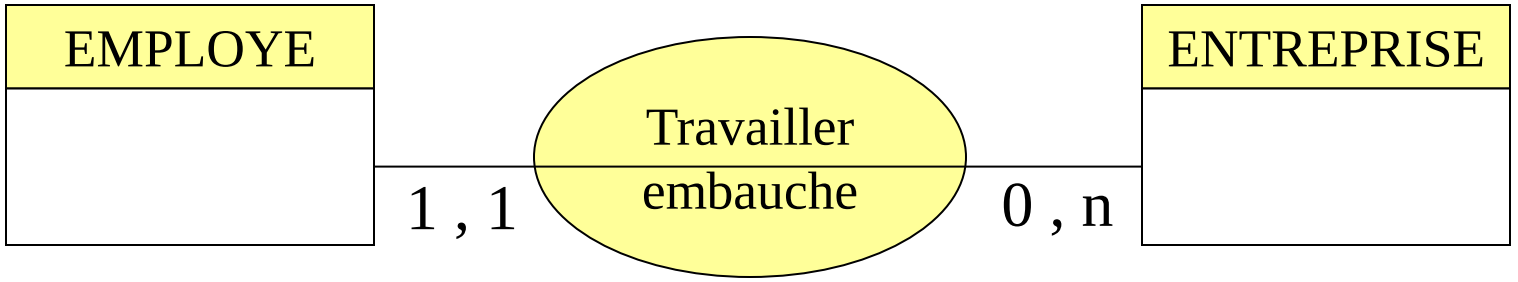


- Un/une employé peut travailler dans 0 à 3 entreprises.
- A l ’inverse, une entreprises peut employer entre 0 et n employés.

  - La cardinalité d’une association pour une entité est constituée d’une borne minimale et d’une borne maximale :

- **Conceptuel : le modèle entité Association E/A (E/R)**
  - *Cardinalité : bornes minimales et maximales*
    - **Minimale** : nombre minimum de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 0 ou 1
      - la borne minimale exprime les contraintes d 'intégrité
    - **Maximale** : nombre maximum de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 1 ou n
      - la borne maximale est nécessaire pour la conception de la base de donnée.

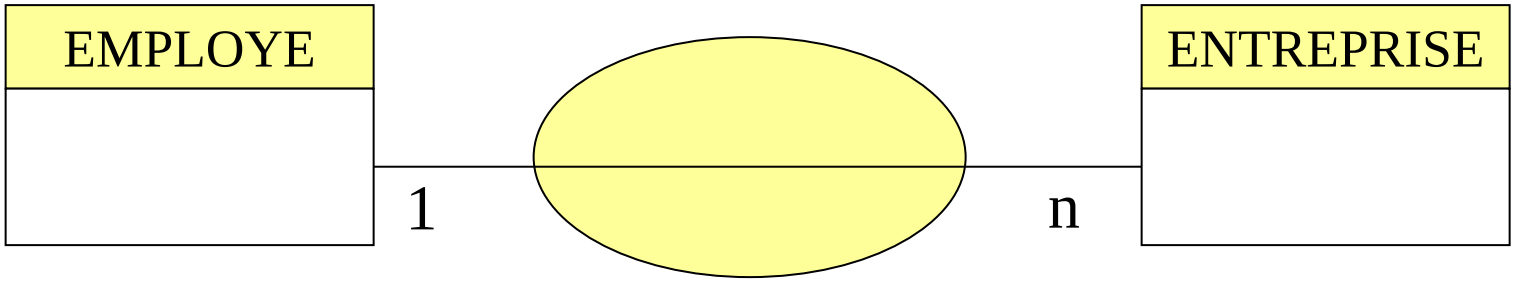
- **Conceptuel : le modèle entité Association E/A (E/R)**
  - *Des cardinalités vers les **Liens**:*



En notant les cardinalités maximales, nous déterminons 3 profiles de **liens** :

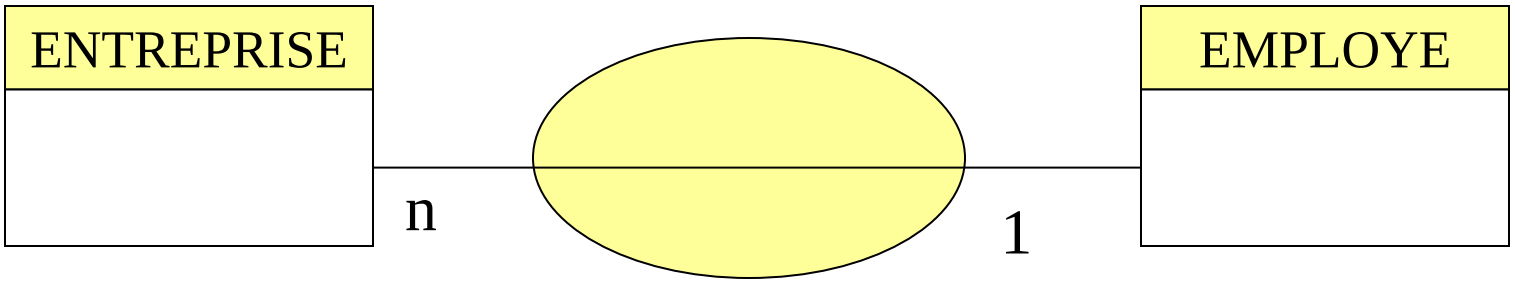
- **fonctionnel** 1:n
- **hiérarchique** n:1
- **maillé** n:m

- **Conceptuel : le modèle entité Association E/A (E/R)**
  - Les Liens : représente le type de lien entre les entités.
    - **Fonctionnels**



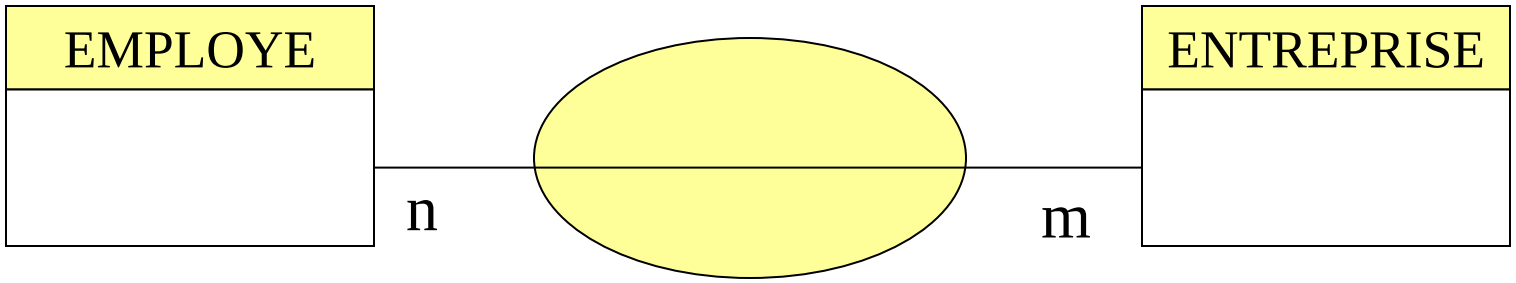
- Une instance de EMPLOYE ne peut être associée qu'à une instance de ENTREPRISE dans cet exemple.

- **Conceptuel : le modèle entité Association E/A (E/R)**
  - **Liens hiérarchiques**
  - représente le lien entre les entités.



  - Une instance de ENTREPRISE peut être associée à plusieurs instances de EMPLOYE dans cet exemple.

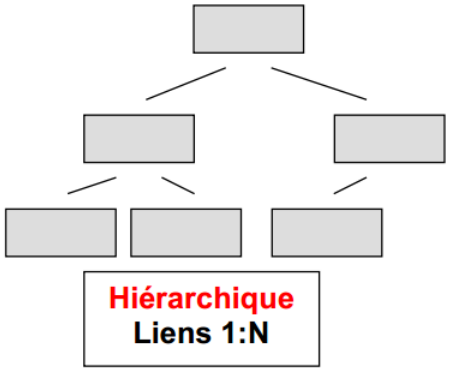
- **Conceptuel : le modèle entité Association E/A (E/R)**
  - Les **liens maillés**
    - représente le lien entre les entités.



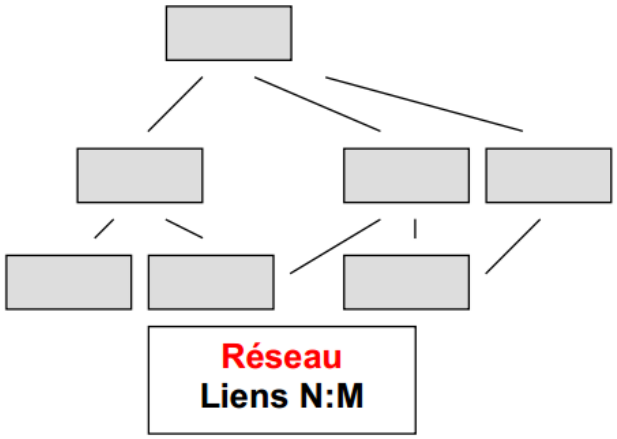
    - Une instance de EMPLOYE peut être travailler dans plusieurs ENTREPRISE.
  - **Résumé de l 'objectif de ce modèle E/A:**
    - la détection d 'erreur de conception avant développement
    - la traduction automatique dans un modèle logique
  - Il est supporté par les outils de type CASE /Rational ROSE.
  - *Il correspond à la partie modélisation de données dans UML .*

- **Schémas logiques :Modèles de bases de données**
- **Les déclinaisons des schémas conceptuels vers le logique**

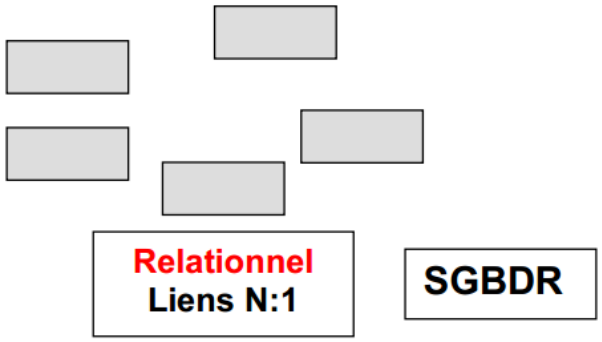
Le modèle arbre



Le modèle graphe



Le modèle relationnel



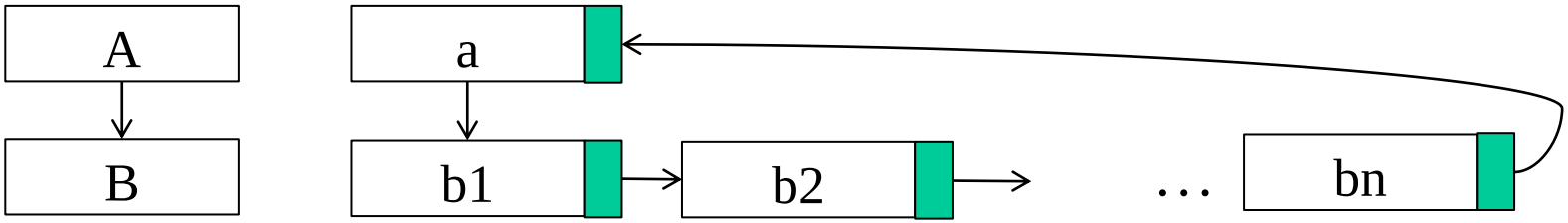
- **Schéma logique : Modèles**

Modèles issus de GRAPHES (hiérarchiques et réseaux)

- Schéma logique organisé sous forme de « graphe »
- Accès au données par navigation ou adressage par liens de chaînage.
- **Modèle réseau** :

  schéma avec nœud et arc : « diagramme de Bachman »

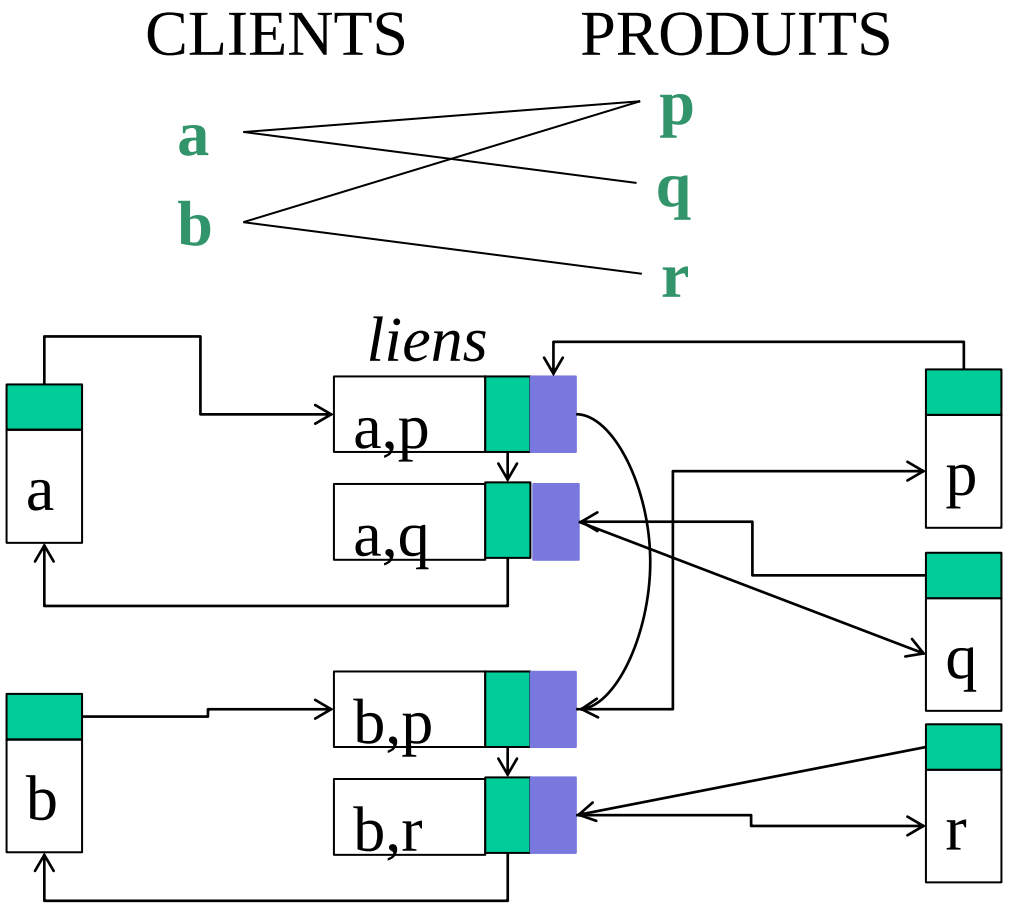
- Implémentation d'un lien par une liste circulaire



- **Schéma logique : Modèles**

Modèles issus de GRAPHES (hiérarchiques et réseaux)

– Exemple de modèle **Réseau** : schéma d'une association N:M par 2 liens CODASYL(voir historique).

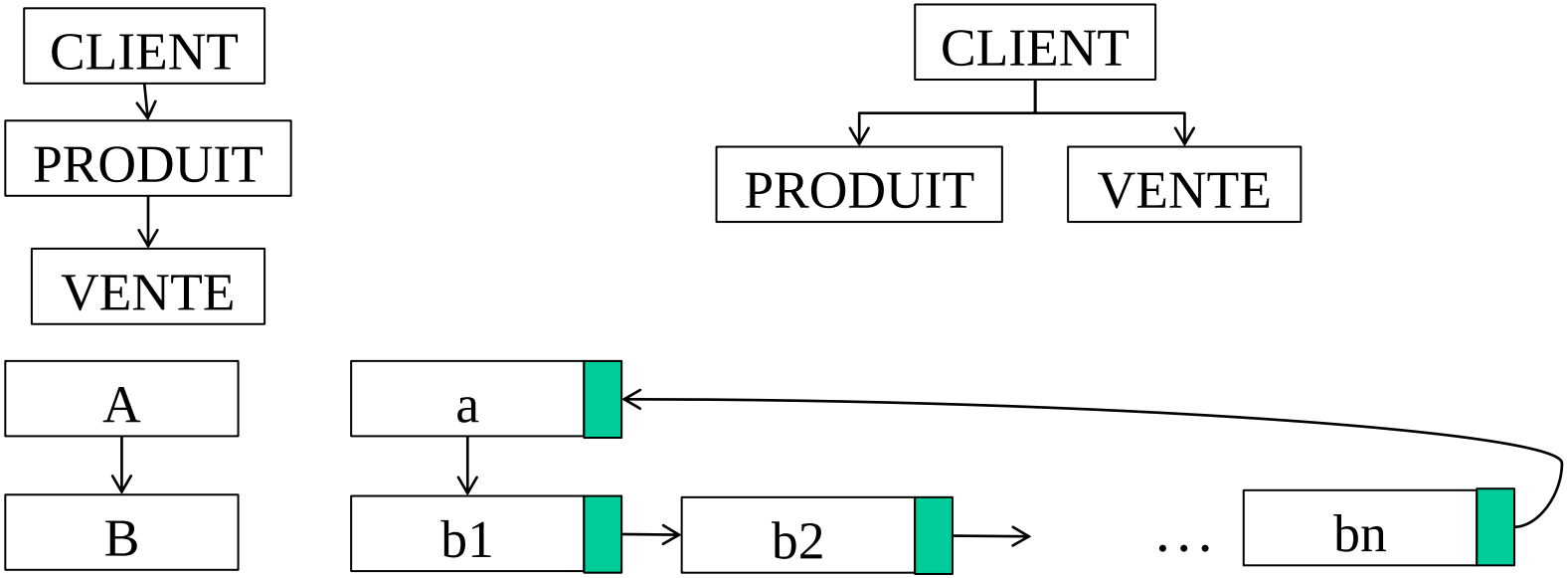


• Schéma logique : Modèles

Modèles issus de GRAPHES (hiérarchiques et réseaux)

– **Modèle hiérarchique** :

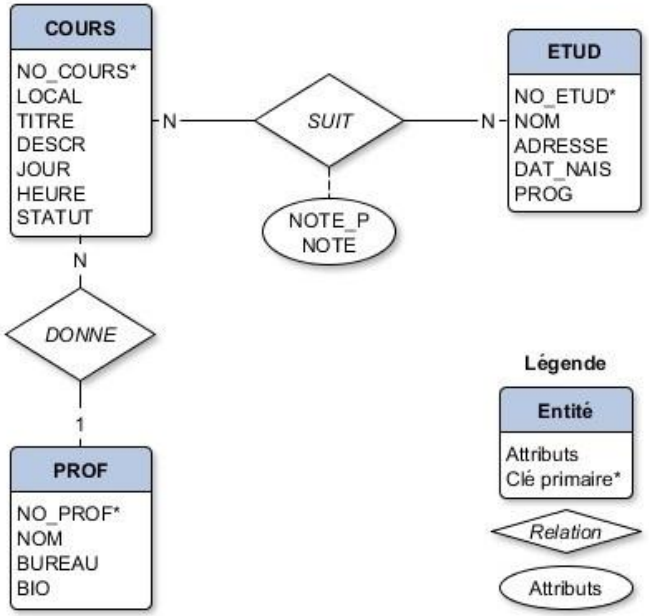
- schémas avec nœud (segment) et arc (lien hiérarchique).
- Plusieurs choix possibles d'arborescences
- Dissymétrie des traitements selon le choix de la racine, ex : lister les produits achetés par le client a ou lister les clients qui ont achetés le produit b2…
- Adapté aux organisations à structure arborescente



- **Schéma logique : Modèles**

Modèle Relationnel

- imaginé par CODD
- Fondé sur la notion mathématique de Relation
- Données organisées en tables avec adressage relatif
- Accès par adressage relatif géré par le SGBD.
- Stratégie d'accès gérée par le SGBD



III LE MODELE RELATIONNEL

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- **Schéma logique : Modèle Relationnel**
- 1970,
  - CODD présente le modèle relationnel
  - Schéma logique représenté par des RELATIONS
- LE SCHÉMA RELATIONNEL
  - C'est l'ensemble des RELATIONS qui modélisent le monde réel
  - Les relations représentent les **entités** du monde réel (des personnes, des objets, etc.) ou les **associations** entre ces entités.
  - Passage d'un schéma conceptuel E-A à un schéma relationnel
    - Une **entité** est représentée par la relation :
      - nom_de_l'entité (liste des attributs de l'entité)
    - une **association** M:N est représentée par la relation :
      - nom_de_l'association (liste des identifiants des entités participantes, liste des attributs de l'association)

    Exemples :

    - CLIENT (IdCli, nom, ville) Entité
    - PRODUIT(IdPro, nom, prix, qstock) Entité
    - VENTE (IdCli, IdPro, date, qte) Association

- Passage d'un schéma E/A (E/R) à un schéma relationnel
  - En définissant les **entités**

Nom (**clé1**,….,**cléN**,attribut1, attribut2, attribut3,……., attributM)

Ou

Nom (clé1,….,cléN,attribut1, attribut2, attribut3,……., attributM)

Ou

NomAssociation **(liste des identifiants,**
liste des attributs de l'association**)**

  - En définissant les **associations** N,M

NomAssociation **(liste des identifiants des entités participantes,**
liste des attributs de l'association**)**

LES AVANTAGES DU MODELE RELATIONNEL LOGIQUE

- Simplicité de présentation par tables
- Opérations relationnelles
  - algèbre relationnelle (plus, moins, divise, mulitplie, etc…)
  - langage « assertionnel » ou spécialisé
- Indépendance physique
  - optimisation des accès
  - stratégie d'accès déterminée par le système
- Indépendance logique
  - concept de VUES
- Maintien de l'intégrité
  - contraintes d'intégrité définies au niveau du schéma

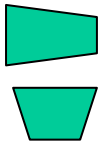
- LES AVANTAGES DU MODELE RELATIONNEL LOGIQUE
- Domaine
  - Ensemble de valeurs atomiques d 'un type sémantique :
    - NOM_SITE = {ATOMIUM, TOUR_EFFEL,STATUE_LIBERTE,...}
    - ensemble des valeurs possibles
- Relation
  - Sous ensemble du produit cartésien de plusieurs domaines
- N-uplets
  - un élément d 'une relation est un n-uplet de valeurs (tuple in english)
- Attributs
  - son nom doit être porteur de sens
  - différent du nom de domaine
  - plusieurs attributs peuvent avoir le même nom de domaine
- Schéma d 'une relation

- Exemple de définition de quatre entités relationnelles
  - Station (**id**, nom, capacité, lieu, région, tarif)
  - Activité (**idStation, libellé**, prix)
  - Client (**id**, nom, prénom, ville, région, solde)
  - Séjour (**id**, *idClient*, *idStation*, début, fin, nbPlaces)

- Entité exprimée sous forme de table
  - exemple de la table Station

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
sa	Santalba	150	Martinique	Antilles	2000
pa	Passac	400	Alpes	Europe	1000

- Les Opérateurs de base de l'algèbre relationnel
  - La restriction ou sélection: σ (unaire)
  - La projection: π (unaire)
  - Le produit cartésien: × (binaire)
  - L'union: ∪ (binaire)
  - L'intersection ∩ (binaire)
  - La division / (binaire)
  - La différence: − (binaire)



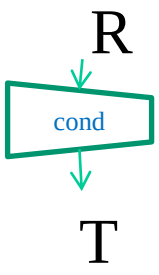
- Opérateurs dérivés
  - *Jointure* : ⋈, la composition d'un produit cartésien et d'une sélection.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- Restriction/sélection
  - But :
    - sélectionner certains N-uplets
    - Réduire la taille verticalement
  - Contraintes
    - Unaire
    - Spécifier une condition
  - Exemple : stations aux Antilles
  - Station_antilles<- $\sigma_{\text{région='Antilles'}}$(Station)
  - Notation textuelle T <- σ_{cond}(R)
  - Notation graphique

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
sa	Santalba	150	Martinique	Antilles	2000
pa	Passac	400	Alpes	Europe	1000

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
sa	Santalba	150	Martinique	Antilles	2000



INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- Projection (unaire)
  - But :
    - sélectionner certains attributs
    - Réduire la taille horizontalement
    - Supprime les doublons
  - Contraintes
    - Unaire
    - Spécifier une liste d'attributs

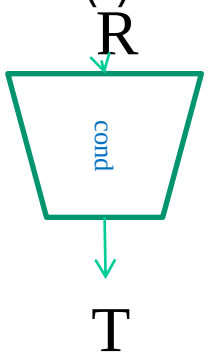
id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
sa	Santalba	150	Martinique	Antilles	2000
pa	Passac	400	Alpes	Europe	1000

Exemple : région des noms

- Station_Region <- π nom,région (Station)

nom	région
Venusa	Antilles
Farniente	Océan Indien
Santalba	Antilles
Passac	Europe

- Notation textuelle T <- π condition (R)
- Notation graphique



- exemple π region(Station)

- Union : (Supprime les doublons)

StationAsie (**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
ma	MangaCity	50	Tokyo	Asie	1800
fa	Farniente	200	Seychelles	Océan Indien	1500

StationEurope (**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
pa	Passac	400	Alpes	Europe	1000
vt	Val Thorens	350	Alpes	Europe	1200

– Station_AE= StationAsie U StationEurope

id	nom	capacité	lieu	région	tarif
pa	Passac	400	Alpes	Europe	1000
vt	Val Thorens	350	Alpes	Europe	1200
ma	MangaCity	50	Tokyo	Asie	1800
fa	Farniente	200	Seychelles	Océan Indien	1500

- Intersection:

Station_Mer (**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
sa	Santalba	150	Martinique	Antilles	2000

Station_Antilles(**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
sa	Santalba	150	Martinique	Antilles	2000
fo	FortAventure	50	Martinique	Antilles	2200

StationMer_Antilles = Station_Antilles ∩ Station_Mer

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
sa	Santalba	150	Martinique	Antilles	2000

- Produit cartésien:

Station (**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
sa	Santalba	150	Martinique	Antilles	2000
pa	Passac	400	Alpes	Europe	1000

Activité (**libellé**, prix)

libellé	prix
randonnée	350
Plongée	150

Activité_Station (**id**, nom, capacité, lieu, région, tarif, **libelle**, prix)

= Activité (**libellé**, prix) X Station (**id**, nom, capacité, lieu, région, tarif)

Produit cartésien: Résultat

Activité_Station (**id**, nom, capacité, lieu, région, tarif, **libelle**, prix) =

Activité (**libellé**, prix) X Station(**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif	libellé	prix
va	Venusa	350	Guadeloupe	Antilles	1200	Randonnée	350
va	Venusa	350	Guadeloupe	Antilles	1200	plongée	150
fa	Farniente	200	Seychelles	Océan Indien	1500	Randonnée	350
fa	Farniente	200	Seychelles	Océan Indien	1500	plongée	150
sa	Santalba	150	Martinique	Antilles	2000	Randonnée	350
sa	Santalba	150	Martinique	Antilles	2000	plongée	150
pa	Passac	400	Alpes	Europe	1000	Randonnée	350
pa	Passac	400	Alpes	Europe	1000	plongée	150

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

- Différence :

Station_Mer (**id**, nom, capacité, lieu, région, tarif)

id	Nom	Capacité	Lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
fa	Farniente	200	Seychelles	Océan Indien	1500
Sa	Santalba	150	Martinique	Antilles	2000

Station_Antilles(**id**, nom, capacité, lieu, région, tarif)

id	nom	capacité	lieu	région	tarif
va	Venusa	350	Guadeloupe	Antilles	1200
sa	Santalba	150	Martinique	Antilles	2000
fo	FortAventure	50	Martinique	Antilles	2200

Station_MerSaufAntilles = Station_Mer - Station_Antilles

id	nom	capacité	lieu	région	tarif
fa	Farniente	200	Seychelles	Océan Indien	1500

Station_MerSaufAntilles = Station_Antilles - Station_Mer

id	nom	capacité	lieu	région	tarif
fo	FortAventure	50	Martinique	Antilles	2200

- Division : (Élimine les doublons)

Station_Nom (nom, capacité, lieu)

Nom	Capacité	Lieu
Venusa	350	Guadeloupe
Venusa	350	Seychelles
Santalba	150	Martinique
Madinia	50	Martinique

Station_Capacité (capacité, lieu)

capacité	lieu
350	Guadeloupe
350	Seychelles

Station_Divise = Station_Nom / Station_Capacité

nom
Venusa

- Jointure :

Station_Nom (nom, capacité)

Nom	Capacité
Venusa	350
Farniente	200
Santalba	400

Region_Lieu (region, lieu)

Région	lieu
Antilles	Guadeloupe
Ocean Indien	Seychelles
Antilles	Martinique

Station_Jointe = Station_Nom ⋈ Region_Lieu (capacité<400 et lieu != Martinique) ou σ capacité<400 et lieu != Martinique (Station_Nom X Region_Lieu)

Nom	Capacité	Lieu	Région
Venusa	350	Guadeloupe	Antilles
Venusa	350	Océan Indien	Seychelles
Farniente	200	Guadeloupe	Antilles
Farniente	200	Océan Indien	Seychelles

- Jointure en conservant une « clé étrangère » et l'intégrité fonctionnelle :

Station_Nom (nom, capacité)

Nom	Capacité
Venusa	350
Farniente	200
Santalba	400

Capacité_Lieu (capacité, lieu)

capacité	lieu
350	Guadeloupe
400	Seychelles
200	Martinique

Station_Jointe = Station_Nom ⋈ Capacité_Lieu (capacité<400 et lieu != Martinique) ou σ capacité<400 et lieu != Martinique (Station_Nom X Capacité_Lieu)

Nom	Capacité	Lieu
Venusa	350	Guadeloupe

- Clé d'une relation
  - primaire :
    - attribut (ou groupe d 'attributs) qui détermine tous les autres
      - PRODUIT (**no_produit**, nom, prix_HT)
    - une clé détermine de manière unique une n-uplet
    - une relation peut posséder plusieurs clés **candidates**
      - *si nom de produit est unique, nom peut devenir la clé primaire.*
  - étrangère ou clé secondaire :
    - attribut (ou groupe d 'attributs) qui fait référence à la clé primaire d 'une autre relation
    - CATEGORIE(**no_categ**, designation, tva)
    - PRODUIT(**no_produit**, nom, marque, *no_categ*, prixHT)

      > no_categ est clé étrangère dans PRODUIT, c'est la clé primaire de CATEGORIE

- Normalisation
  - Afin de réduire les doublons.
  - Définition des formes normales ajoutés au schéma relationnel :
    - **1ère forme : 1FN** : une relation est en 1FN si tout attribut est atomique à savoir non décomposable.

ELEVE(**n_eleve**, nom_prenom, liste_notes) devient en 1FN :

- ELEVE(**n_eleve**, nom, prenom)
- NOTE(**n_eleve**, matiere, note)

- Définition des formes normales ajoutés au schéma relationnel :
  - **3ème forme : 3FN** : une relation est en 2FN si :
    - elle est en 2FN
    - si tout attribut n'appartenant pas à la clé primaire ne dépend pas d'un attribut non clé

    Ou

    - aucun attribut ne faisant pas partie de la clé primaire ne doit dépendre d'une partie des autres attributs ne faisant pas non plus partie de la clé primaire.

PERSONNE (**id_personne**, civilité, nom, prenom, sexe) devient en **3FN** :

- PERSONNE (**id_personne** , civilité, nom, prenom)
- CIVILITE (**civilité**, sexe)

- Définition des formes normales ajoutés au schéma relationnel :
  - **2ème forme : 2FN : une relation est en 2FN si :**
    - **elle est** en 1FN
    - tout attribut n'appartenant pas à la clé ne dépend pas que d'une partie de la clé ou
    - aucun attribut ne faisant pas partie de la clé primaire ne doit dépendre que d'une partie de la clé primaire.

COMMANDE(**date**, **n_cli**, **num_produit**, qte, prix_unitaireHT)

Le prix_unitaire HT est dépendant de n°produit : partie de la clé !

Manière de le mettre en 2FN

- COMMANDE (**date**, **n_cli**, **num_produit**, qte)
- PRODUIT(**num_produit**, prix_unitaire_HT)

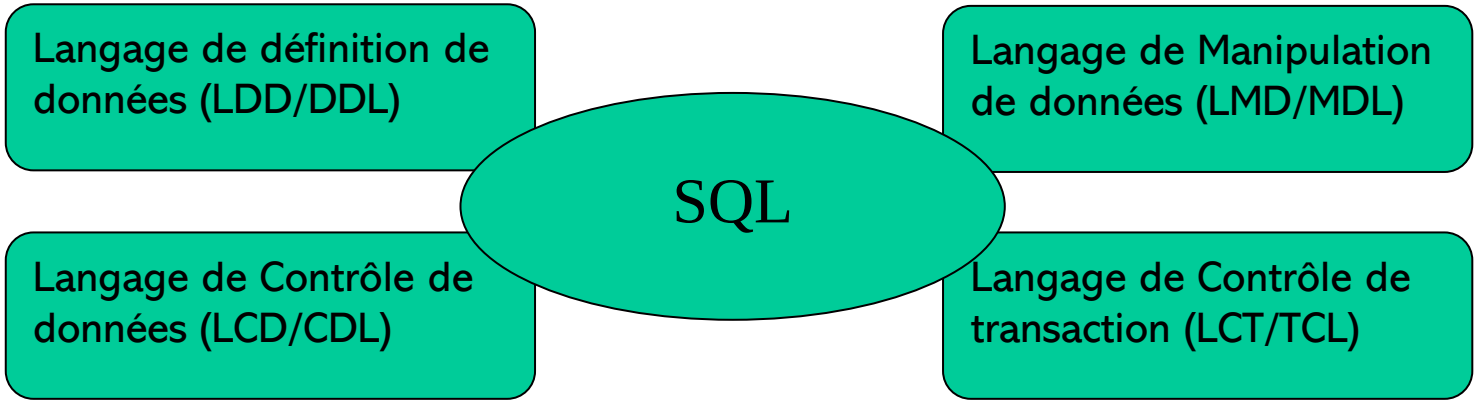
! Ne pas respecter la 2FN entraîne des redondances. Cela gaspille de l'espace de stockage, et pose aussi le problème de la mise à jour des données.

- Intégrité
  - de domaine
    - les valeurs d 'une colonne de relation doivent appartenir au domaine correspondant (*liste de valeurs possibles*)
      - contrôle des valeurs des attributs
      - contrôle entre valeurs des attributs
  - de clé
    - les valeurs de clés primaires doivent être uniques
      - uniques => unicité de clé
      - non nulles => unicité des n-uplets
  - référentielle
    - les valeurs de clés étrangères sont nulles ou sont des valeurs de la clé primaires auxquelles elles font référence
      - relations indépendantes

      Séjour (**id**, *idClient*, *idStation*, début, fin, nbPlaces)

IV LE LANGAGE SQL

- Structured Query Langage (IBM)
  - Issu du langage QUEL, SEQUEL commercialisé par ORACLE
  - Inclut une grande partie des possibilités de l'algèbre relationnel
  - langage interprété : interface d 'utilisation (MySql, Squirrel, SQLylog, Dbeaver, MSAccess,…)
  - langage intégré dans un programme (C, JAVA) : interface de programmation
  - procédures exécutés dans le SGBDR Intégrés mais exécutable de l 'extérieur par les interfaces d 'utilisation ou de programmation.



- Structured Query Langage (IBM):
  - Les commandes SQL
    - Définition de données (LDD)

      CREATE

      DROP

      ALTER
    - Manipulation de données (LMD)

      SELECT

      INSERT

      UPDATE

      DELETE
    - Contrôle de transactions (LCD)
      - Accès concurrents
        - COMMIT
        - ROLLBACK
      - Droits d'accès
        - GRANT / REVOKE

```
--LDD --
CREATE TABLE table
(
-- définition des colonnes
Nom  et type de colonne [ NOT NULL [UNIQUE] ] [ DEFAULT valeur ]
       [ PRIMARY KEY ] [ REFERENCES table ] [ CHECK condition ] ,... ,
-- contraintes de table
    [ PRIMARY KEY (liste de colonnes) ],
    [ UNIQUE (liste de colonnes) ] ,... ,
    [ FOREIGN KEY (liste de colonnes) REFERENCES table
    [ ON DELETE {RESTRICT | CASCADE | SET NULL} ]
    [ ON UPDATE {RESTRICT | CASCADE | SET NULL} ] ,... ,
    [ CHECK condition ] ,...
)
```

-- Exemple de LDD--

```
CREATE TABLE vente
    (
        IdCli     CHAR(4)   NOT NULL          REFERENCES client        ,
        IdPro     CHAR(6)   NOT NULL,
        date      DATE      NOT NULL,
        qte       SMALLINT                    CHECK (qte BETWEEN 1 AND 10),
      -- contrainte de table
      PRIMARY KEY (IdCli, IdPro, date)                ,
      FOREIGN KEY (IdPro) REFERENCES produit
      --ON DELETE CASCADE  ON UPDATE CASCADE  --SQL2 ONLY
    )
```

- ALTER TABLE permet de modifier la table

  ALTER TABLE client ADD COLUMN telephone CHAR(16)

- DROP TABLE détruit immédiatement l'accès la table

  DROP TABLE client

- CREATE INDEX permet d 'accélérer les recherches et de créer des indexes multi-colonnes

  CREATE [UNIQUE] INDEX index

  ON table (colonne [ASC|DESC], ...)

  l 'option UNIQUE assure l 'unicité de la clé

  l 'option ASC(endant) ou DESC(endant) permet de connaître l 'ordre de création de l 'indexe et donc de la recherche.

- Types de données
  - chaînes de caractère
    - **CHAR**(n) chaîne constante ou n est la longueur maximale
    - **VARCHAR**(n) chaîne variable ou n est la longueur maximale
  - entiers
    - **SMALLINT** sur 2 octets -32.768 à +32.767
    - **INTEGER** sur 4 octets -2.147.483.648 à +2.147.483.647
  - décimaux
    - **NUMERIC** => (n,m) le nombre de décimal doit être exactement m
    - **DECIMAL**(n,m)

- Types de données
  - Numériques à virgule flottante
    - **REAL** (SIMPLE PRECISION) au moins 7 chiffres significatifs
    - **FLOAT** (DOUBLE PRECISION) au moins 15 chiffres significatifs
  - Types temporels (https://www.data-transitionnumerique.com/sql-date/)
    - **DATE** : **YYYYMMDD** - jour : 2 chiffres, mois : 2 chiffres, année : 4 chiffres
    - **TIME** : HHMMSS,MMM - heures, minutes, secondes, millièmes:
    - **TIMESTAMP** : heures, minutes, secondes : ex : **1677657600**
      - Wed Mar 01 2023 09:00:00 UTC+0100
    - **INTERVAL** : intervalle de temps (INTERVAL 10 YEARS)
  - Valeur NULL
    - colonne non renseignée et donc vide d'information. La valeur n'est pas zéro, c'est une absence de valeur

- Contraintes d'intégrité
  - NOT NULL valeur null impossible
  - UNIQUE unicité d'un attribut
  - PRIMARY KEY clé primaire
  - FOREIGN KEY clé étrangère
  - CHECK plage ou liste de valeurs
- Toute opération violant une des contraintes sera rejetée
- Le système garantit l'intégrité des données
- Les contraintes d'intégrité peuvent être ajoutées ou supprimées :
  - sur une table
  - sur une colonne

Exemple

```
ALTER TABLE SALARIE
DROP CONSTRAINT NOM_UNIQUE
ADD CONSTRAINT SAL_MIN CHECK(SAL > 1000)
RENAME CONSTRAINT NOM1 TO NOM2
```

--Manipulation de données (LMD) -- 4 commandes : SELECT INSERT UPDATE DELETE

- SELECT : selection de lignes ou colonnes

```
SELECT P.prix
FROM produit P
WHERE P.idPro = 'p1'
```

- INSERT ajout de lignes,

```
INSERT INTO client
(idPro, nom, ville)
VALUES ('c42','Duchemin','Bourges')
```

- UPDATE mise à jour de lignes,

```
UPDATE TABLE produit
SET P.prix= P.prix*1,21
WHERE  P.idPro='p2'
```

- DELETE suppression de lignes.

```
DELETE FROM produit
WHERE P.idPro='p2'
```

--Manipulation de données (LMD) – SELECT

SELECT	[DISTINCT] liste d'attributs, expressions, agrégat
FROM	liste de tables ou vues
WHERE	clause de recherche : qualifications / prédicats
GROUP BY	liste d'attributs de partion.
HAVING	qualification de groupe (agrégat)
ORDER BY	liste de colonnes [ASC | DESC]

- Contrairement à l'algèbre relationnel, SQL n'élimine pas les doublons : pour cela, il faut spécifier DISTINCT !
- les opérations arithmétiques sont disponibles (+, -, *, /)
- les opérateurs d 'agrégats COUNT, MIN, MAX, SUM, AVG
- l 'étoile permet de lister tous les attributs
- Peut utiliser un tri du résultat ORDER
- Peut découper le résultat selon certains critères GROUP BY
- Peut restreinte le résultat selon certains critères HAVING

- La clause WHERE : condition de recherche
  - une condition de recherche est spécifiée par un prédicat.
  - Les qualifications ou prédicats simples :
    - (=, <>, <, >, <=, >=)
    - **LIKE** contient ‘ %toto% ’ commence par ‘ toto% ’ , finit par ‘ %toto ’
    - **BETWEEN** : colonne entre deux valeurs : (BETWEEN 5000 AND 12000)
    - **IS NULL** : dont la valeur est inconnue
    - **IN** : colonne dans une liste :
      - P.marque IN (‘ LENOVO ’, ‘ APPLE ’, ‘ DELL ’)
    - **EXIST** : au moins une valeur définie
      - SELECT * FROM R
        WHERE EXISTS (SELECT * FROM P WHERE R.attribut=P.attribut)
    - **ALL** : prédicat vrai pour tous
    - **ANY** : prédicat vrai pour au moins un

- La clause WHERE : condition de recherche
  - Les prédicats composés
    - composé de plusieurs prédicats simples articulés par :
      - **AND**
      - **OR**
      - **NOT**

- Les opérateurs d'agrégats
  - Pas d'équivalent en algèbre relationnelle,
  - Permettent d'effectuer des calculs sur des groupes de n-uplets
  - Principaux opérateurs :
    - **COUNT**: nombre de valeurs ou n-uplets,
    - **MIN** : minimum des valeurs,
    - **MAX** : maximum des valeurs,
    - **SUM** : somme des valeurs,
    - **AVG** : moyenne des valeurs.
- Exemple :

Nombre de client : SELECT COUNT(*) FROM client

Nombre de produit : SELECT COUNT(DISTINCT no_pro) FROM produit

- Le tri du résultat d ’un SELECT
  - **ORDER BY** : spécifie les colonnes qui vont définir le critère de tri dans l’ordre des colonnes
  - **ASC** (croissant) **, DESC** (décroissant): l ’ordre de tri ORDER BY peut être précisé par ASC ou DESC. Par défaut ASC est utilisé.

- **HAVING** Spécifie une condition de restriction de groupe nécessite la présence de l ’opérateur **GROUP BY**

```
SELECT      P.marque, AVG ( P.prix )
FROM        produit P
GROUP BY    P.marque
HAVING      AVG ( P.prix ) < 5000
```

- LES VUES
  - Une vue est une vision partielle ou particulière des données d'une ou plusieurs tables de la base.
  - La définition d'une vue est donnée par un SELECT qui indique les données de la base qui seront vues.
  - Les données des tables peuvent être modifiées à travers la vue
  - la commande de création de vue

    CREATE VIEW V_R (col1, col2...) AS SELECT … FROM R….
  - La spécification des noms des colonnes de la vue est facultative : par défaut, les colonnes de la vue ont pour nom les noms des colonnes résultat du SELECT.
  - Le SELECT d ’une vue ne peut pas utiliser la clause ORDER BY.
  - la commande de suppression de vue

    DROP VIEW V_R

- LCD : Langage de contrôle des données
  - Accès concurrents
    - COMMIT : valider la transaction
    - ROLLBACK : la transaction est annulée
  - Droits d'accès
    - GRANT : accorder des privilèges à d 'autres utilisateurs ou à tous
    - REVOKE : supprimer des privilèges à d autres utilisateurs
      - pour les droits de SELECT, INSERT, UPDATE, DELETE
  - Exemple : On accorde à Morgan le droit de sélectionner et d'insérer des lignes dans la relation Etudiant. Un utilisateur ayant reçu ce privilège peut le transmettre à son tour.

GRANT SELECT ON Etudiant TO Morgan With Grant Option

ou à tous : GRANT SELECT ON Etudiant TO **PUBLIC** With Grant Option

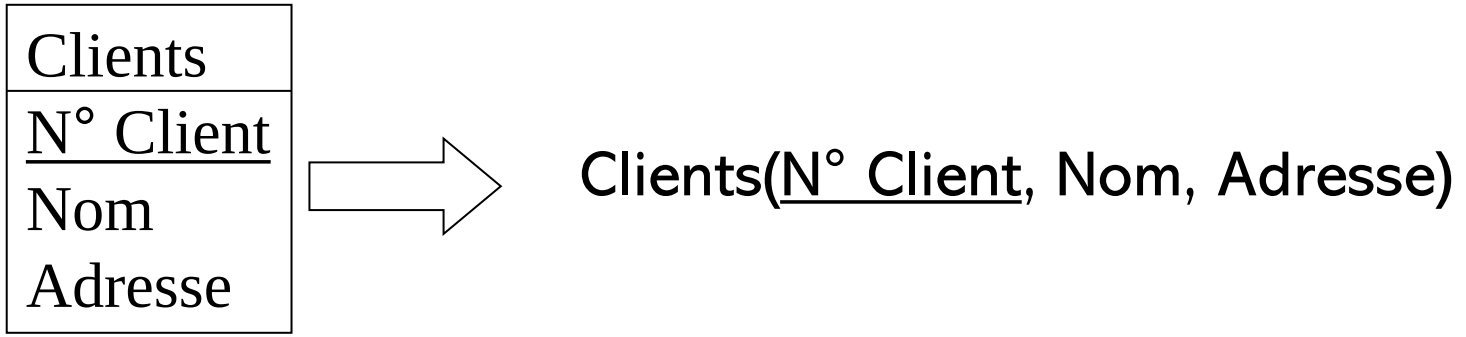
et la suppression :

REVOKE privilège ON Etudiant FROM Tarik

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

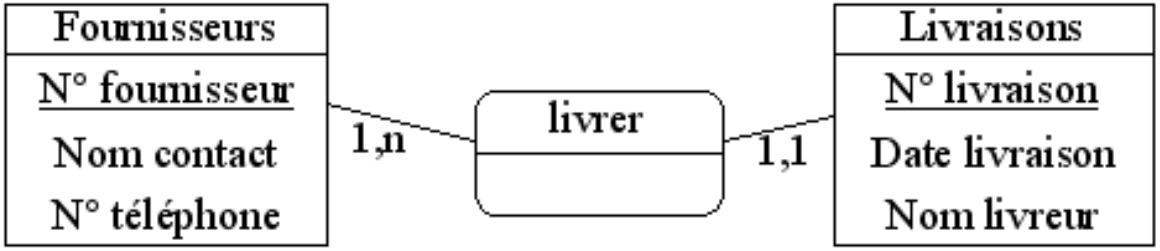
- Pour traduire un Modèle Conceptuel de Données en Modèle Relationnel de Données il suffit d'appliquer **5 règles**.
- Notations : on dit qu'une association binaire (entre deux entités ou réflexive) est de type :
  - 1 : 1 (un à un) si aucune des deux cardinalités maximales n'est de n,
  - 1 : n (un à plusieurs) si une des deux cardinalités maximales est de n,
  - n : m (plusieurs à plusieurs) si les deux cardinalités maximales sont de n.
- Un schéma relationnel ne peut pas faire la différence entre 0,n et 1,n. Par contre, il peut la faire entre 0,1 et 1,1.
- **Règle n°1 :** toute entité devient une table dans laquelle les attributs deviennent les colonnes. L'identifiant de l'entité constitue la clé primaire de la table.



TRADUCTION du MCD vers un LMD Relationnel..

Règle n°2 : une association binaire de type 1 : n disparaît au profit d'une clé étrangère dans la table côté 0, 1 ou 1,1 qui fait référence à la clé primaire de l'autre table. Cette clé étrangère ne peut pas recevoir la valeur vide ou NULL, si la cardinalité est **1, 1**.

Exemple :



L'association livrer de la figure précédente se traduit par :

Fournisseurs(N° fournisseur, nom contact, n° téléphone),

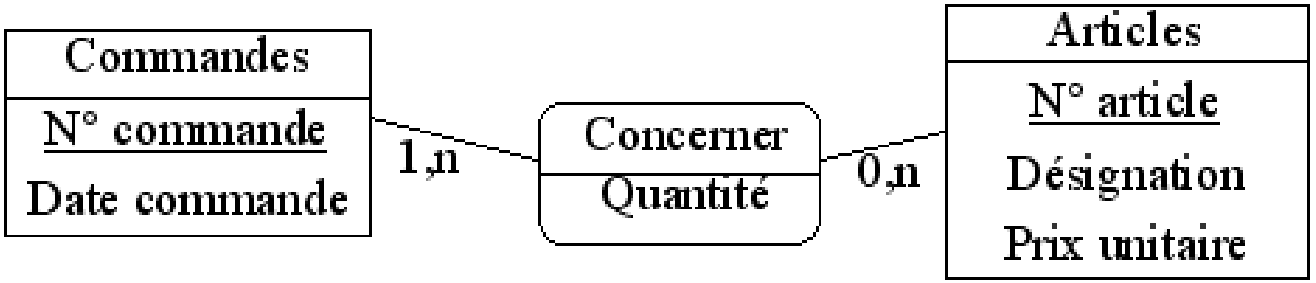
Livraisons(N° livraison, date livraison, nom livreur, #N° fournisseur (non vide)).

Il ne devrait pas y avoir d'attribut dans une association de type 1 : n, mais s'il en reste ils glissent vers la table côté 1:1.

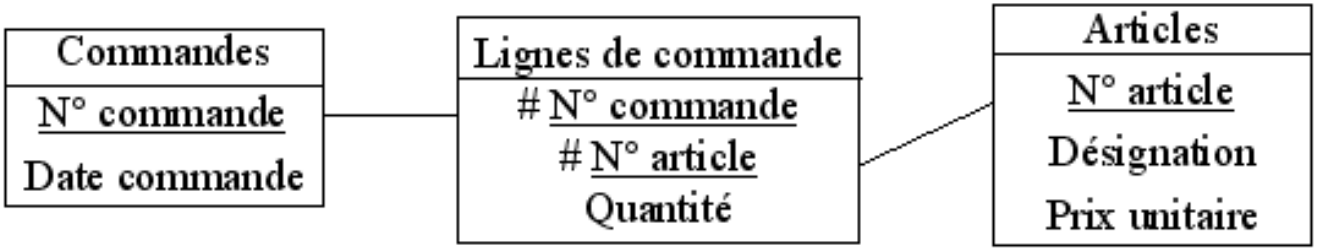
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

- **Règle n°3 :** une association binaire de type n : m devient une table supplémentaire (parfois appelée table de jonction, table de jointure ou table d'association) dont la clé primaire est composée de deux clés étrangères (qui référencent les deux clés primaires des deux tables en association). Les attributs de l'association deviennent des colonnes de cette nouvelle table
- Exemple



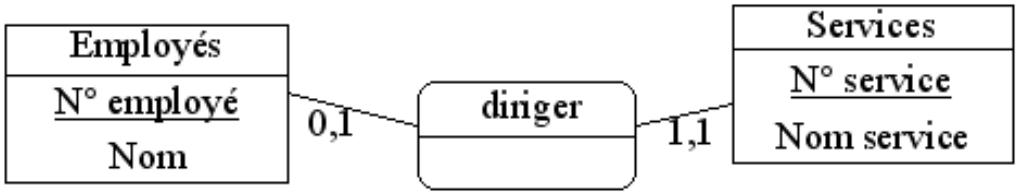
- Traduction



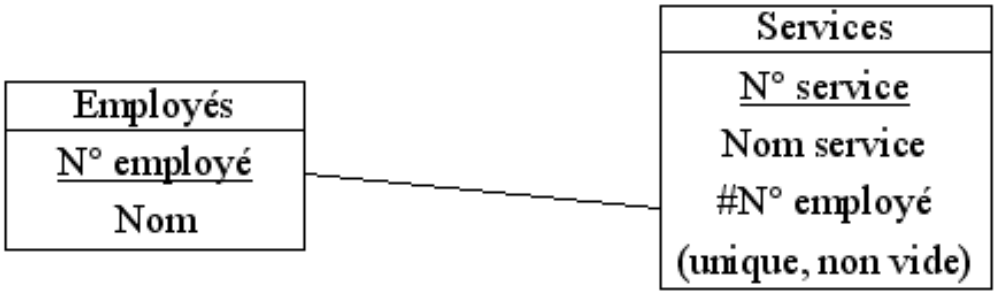
TRADUCTION du MCD vers un LMD Relationnel.

Règle n°4 : une association binaire de type 1 : 1 est traduite comme une association binaire de type 1 : n sauf que la clé étrangère se voit imposer une contrainte d'unicité en plus d'une éventuelle contrainte de non vacuité (cette contrainte d'unicité impose à la colonne correspondante de ne prendre que des valeurs distinctes).

Il devrait y avoir au moins un côté de cardinalité 0,1. C'est alors dans la table du côté opposé que doit aller la clé étrangère. Exemple



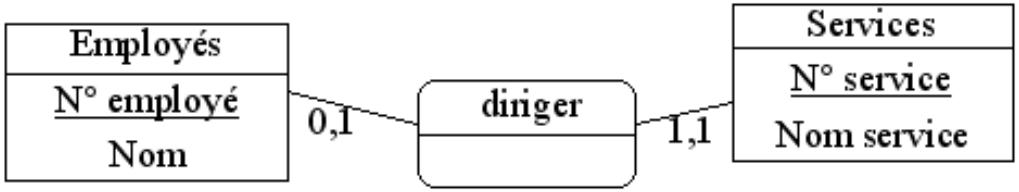
Traduction



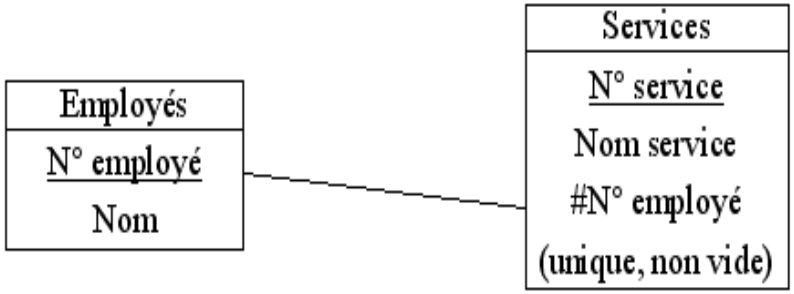
TRADUCTION du MCD vers un LMD Relationnel.

Règle n°4 – suite : Si les deux côtés sont de cardinalité 0,1 alors la clé étrangère peut être placée indifféremment dans l'une des deux tables

Exemple



Traduction

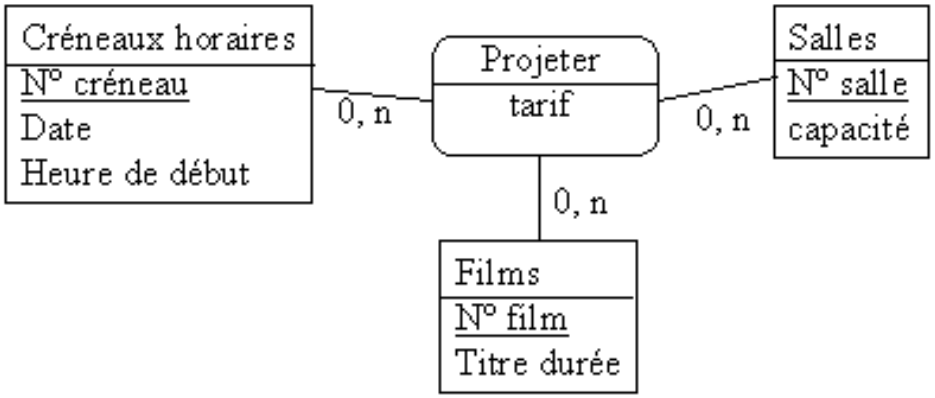


INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

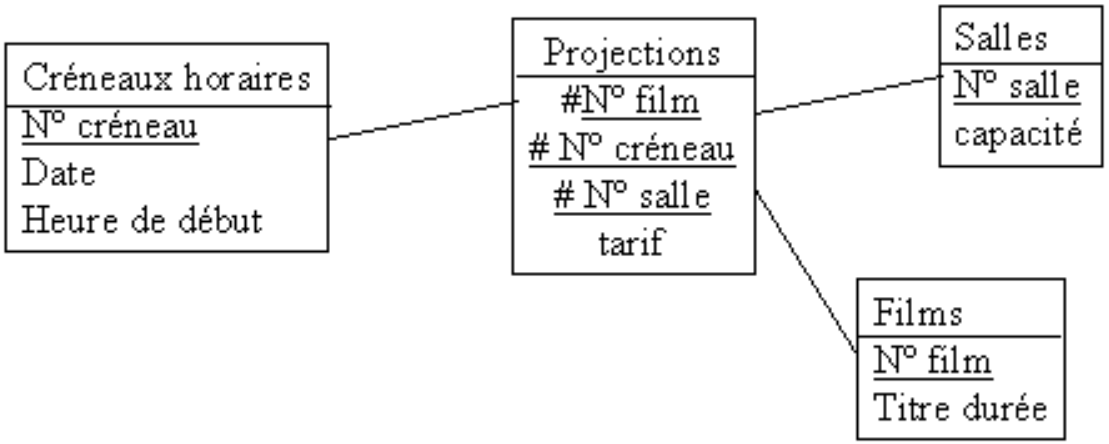
TRADUCTION du MCD vers un LMD Relationnel.

Règle n°5 : une association non binaire est traduite par une table supplémentaire dont la clé primaire est composée d'autant de clés étrangères que d'entités en association. Les attributs de l'association deviennent des colonnes de cette nouvelle table.

Exemple :



Traduction

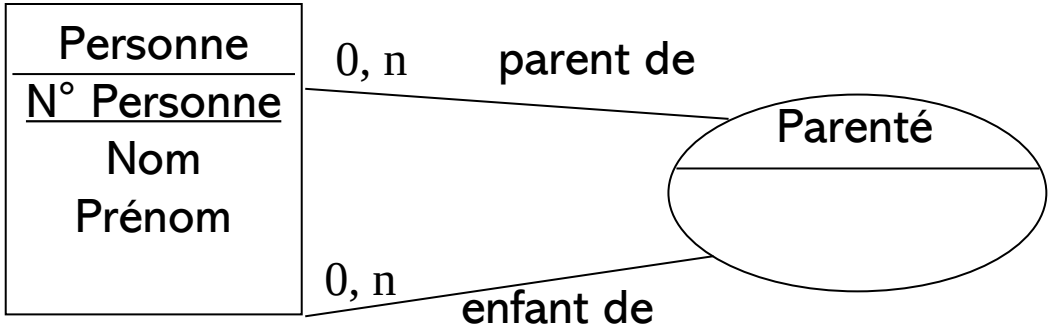


INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

cas de la relation réflexive : Dans certains cas, elle peut se traduire par une relation porteuse de deux attributs, duplications de l'identifiant de l'objet, et toutes deux renommées.

Exemple



Traduction

- Personne(N° Personne, Nom, Prénom),
- Parenté(N° Parent, N°Enfant).

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

- **Selection**
  - SELECT * FROM *R* WHERE *Condition (dateEpreuve >= 01/10/04)*

NoEt	matiere	dateEpreuve	note
12	Maths	04/10/04	12
17	Géographie	07/10/04	9
12	Géographie	07/10/04	15
19	Maths	03/09/04	13
23	Droit	08/09/04	8
17	Droit	08/09/04	14

→

NoEt	matiere	dateEpreuve	note
12	Maths	04/10/04	12
17	Géographie	07/10/04	9
12	Géographie	07/10/04	15

- **Projection** :
  - SELECT DISTINCT(attribut) FROM LesRésultats WHERE Condition (NoEt>16)

NoEt	matiere	dateEpreuve	note
12	Maths	04/10/04	12
17	Géographie	07/10/04	9
12	Géographie	07/10/04	15
19	Maths	03/09/04	13
23	Droit	08/09/04	8
17	Droit	08/09/04	14

→

NoEt
17
23

TRADUCTION du MCD vers un LMD Relationnel.

- **Renommage d 'une relation – exemple sur le produit cartésien**
  - un alias peut désigner une relation, il est valide pour la requête courante
  - il permet de simplifier la lecture d 'une expression
  - ex :

SELECT DISTINCT (NoEt, nomEnseignant)

FROM LesInscription R1,LesEnseignants R2

WHERE R1.matiere = R2.matiere

NoEt	matiere
12	Maths
17	Géographie
12	Géographie
19	Maths
23	Droit
17	Droit
12	Droit
17	Maths

matiere	nomEnseignant
Maths	Dupond
Géographie	Durand
Droit	Leblanc

→

NoEt	nomEnseignant
12	Dupond
17	Durand
12	Durand
19	Dupond
23	Leblanc
17	Leblanc
12	Leblanc
17	Dupond

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

- **Traduction du produit cartésien**
  - SELECT * FROM R,S
  - ex :

```
SELECT DISTINCT ( R1.NoEt, R2.NoEt)
FROM LesResultats R1, LesResultats R2
```

NoEt	matiere	dateEpreuve	note
12	Maths	04/10/04	12
17	Géographie	07/10/04	9
12	Géographie	07/10/04	15
19	Maths	03/09/04	13
23	Droit	08/09/04	8
17	Droit	08/09/04	14

→

NoEt	NoEt
12	12
12	17
12	19
12	23
17	12
17	17
17	19
17	23
19	12
19	17
19	19
19	23
23	12
23	17
23	19
23	23

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

- **Traduction de la jointure**
  - SELECT * FROM R,S WHERE Condtion
  - ex :

SELECT DISTINCT (NoEt, nomEnseignant)

FROM LesInscriptions, LesEnseignants

WHERE LesInscriptions.matiere = LesEnseignants. matière

NoEt	matiere
12	Maths
17	Géographie
12	Géographie
19	Maths
23	Droit
17	Droit
12	Droit
17	Maths

matiere	nomEnseignant
Maths	Dupond
Géographie	Durand
Droit	Leblanc

NoEt	nomEnseignant
12	Dupond
17	Durand
12	Durand
19	Dupond
23	Leblanc
17	Leblanc
12	Leblanc
17	Dupond

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

– Traduction de l'appartenance ∈

```
SELECT * FROM R
WHERE R.attribut IN ( Select attribut FROM S WHERE C )
```

– Traduction de l'exclusion ∉

```
SELECT * FROM R
WHERE R.attribut NOT IN ( Select attribut FROM S WHERE C )
```

TRADUCTION du MCD vers un LMD Relationnel.

- Traduction de l 'Union

SELECT * FROM R WHERE Condition

UNION

SELECT * FROM S WHERE Condition

- Exemple :

SELECT NoEt FROM LesInscriptions WHERE matiere='Maths '

UNION

SELECT NoEt FROM LesInscriptions WHERE matiere = 'Géographie'

NoEt	matiere
12	Maths
17	Géographie
12	Géographie
19	Maths
23	Droit
17	Droit
12	Droit
17	Maths

NoEt
12
17
19

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

- Traduction de l intersection

SELECT * FROM R WHERE Condition

INTERSECT

SELECT * FROM S WHERE Condition

- Exemple :

SELECT NoEt FROM LesInscriptions WHERE matiere='Maths ’

INTERSECT

SELECT NoEt FROM LesInscriptions WHERE matiere = 'Géographie'

NoEt	matiere
12	Maths
17	Géographie
12	Géographie
19	Maths
23	Droit
17	Droit
12	Droit
17	Maths

NoEt
12
17

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

– Traduction de la différence : opérateur -

SELECT * FROM R

MINUS

SELECT * FROM S

Exemple :

SELECT NoEt FROM LesInscriptions WHERE matiere=Droit'

MINUS

SELECT NoEt FROM LesResultats WHERE matiere = 'Droit'

NoEt	matiere
12	Maths
17	Géographie
12	Géographie
19	Maths
23	Droit
17	Droit
12	Droit
17	Maths

NoEt	matiere	dateEpreuve	note
12	Maths	04/10/04	12
17	Géographie	07/10/04	9
12	Géographie	07/10/04	15
19	Maths	03/09/04	13
23	Droit	08/09/04	8
17	Droit	08/09/04	14

NoEt
12

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

TRADUCTION du MCD vers un LMD Relationnel.

– Traduction de la division : Pas d 'équivalent en SQL !

R/S devient

```
SELECT A FROM R
MINUS
( SELECT * FROM R WHERE A,B IN (
            ( SELECT R.A, S.B FROM R,S
                        MINUS
             SELECT A,B FROM R )
)
```

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

V UN APERCU DE « NOSQL »

- La réponse est

- Un effet de mode
  - Le Big Data : un tournevis sonique
  - Le Big Data : la solution UNIVERSELLE
  - Sous le label NoSQL, tout est mis …
  - Le Big Data c 'est nouveau ! Il a plus de 20 ans…

- NOSQL : Not Only Structured Query Langage
  - la fin d'un standard entre les constructeurs
- Une réalité plus nuancée
  - un fichier est déjà une base de données NoSQL…
  - Ne replace pas SQL mais le complète
  - Une base NOSQL, pourquoi pas, mais **quel est le besoin** ?

- Pourquoi faire évoluer les modèles existants
  - Critiques du modèle relationnel
    - **Manque d´évolutivité**
      - les besoins et les données évoluent mais pas le modèle.
      - L 'intégration de données est devenu un secteur d'activité à part entière de l'entreprise.
    - **Manque d'efficacité** La masse des données est multiplié par deux tous les 6 mois (acteurs de type réseaux sociaux,…)
      - informatique décisionnelle pour l'analyse de grands jeux de données (banque et désormais industrie)
      - Big Data pour la gestion et l'analyse de masses de données.

- Une réalité plus nuancée
  - Une vraie avancée avec des projets comme redis, riak, elastic search ou mongodb…. mais une abondance de projets en cours de développement
  - Une vraie perte : chaque projet est dédié par profil d'application…fini la standardisation.
  - Nosql et les schémas relationnels : besoin encore plus fort de réfléchir au modèle pour optimiser les indexes : les indexes sont à la base de la performance. Ils doivent être restreints pour maximiser les performances et minimiser les volumes des indexes.
  - Nosql et l'heure du choix: besoin encore plus fort de réfléchir au besoin pour partir sur le bon projet NoSql : orienté GIS, document, détection temps réelle,…

- Schémas des besoins dans une banque :
  - le système opérationnel va :
    - mémoriser toutes les informations sur les clients, leurs différents comptes et les opérations sur ces comptes : débit, crédit, virement, change, ...
    - déclencher des actions (envoi de relevés, ...)
  - le système décisionnel conserve sur une période (p.e. annuelle)
    - le nombre d'opérations, leur montant moyen, la moyenne des soldes sur les différents comptes, ... sur une durée historique de une ou plusieurs années.
    - Il dispose également d'informations (très, trop) personnelles sur le client…

Exemple Pour préparer un nouveau produit d'assurance vie, donner le nombre de client pouvant être ciblés sur des produits de niveau de risque 4:

- Donner les revenus par tranches d 'âges par mois depuis 2010
- Donner le montant moyen du solde courant sur le compte depuis 5 ans,
- Comparer le type de produits achetés par région et le mode de souscription
- -> ciblage client

Constat lié au besoin : le SQL peut répondre à ce besoin, mais les requêtes sont complexes à écrire et peu lisibles. le temps de calcul peut s 'avérer élevé et nuire au système opérationnel

- Evolution des données
  - **Apparition** de grandes plateformes autour du Web, moteurs de recherche, base de documents type cloud, réseaux sociaux,…
  - **Volume** considérable lié au données du web (videos, photosb..) et aux capteurs (outils connectés, rapport, surveillance réseau,...)
  - Données **évolutives** sans structure connue à l 'avance.

- Evolution des besoins en gestion de données
  - répartir les données pour tenir les performances malgré la masse de données.
  - Nécessité de gérer des données hétérogènes et évolutives

- NoSQL est arrivé par le besoin de répondre aux 3V
  - Volume
  - Vélocité
  - Varieté

- Les 4 grandes familles de « nouveaux » modèles
  - Clés-Valeurs
  - Colonnes
  - Documents
  - Graphes

- Modèle à appliquer
  - Modèle relationnel ?
    - Assurer l 'intégrité dans un contexte multi-utilisateurs et de système opérationnel : gérer les achats de produits, les clients, les portefeuilles.
    - Modèle basé sur des tables (entités) et des jointures (relations) avec l 'application des formes normales.
  - Modèle en étoile ?
    - permettre d 'effectuer des analyses efficaces pour produire des rapports en vue de prise de décision
    - Modèle basé sur la notion de fait (quantité que l 'on souhaite analyser) et de dimension (axe d 'analyse)
    - Exemple
      - » Fait : souscription du produit à un certain prix et pour une certaine quantité.
      - » Dimension : temporelle (toujours vrai) mais aussi spatiale, client, vendeur, type de produit,….
  - Exemple de modèle en étoile
    - FAITACHAT(refjour,refclient, refvendeur, refproduit, refprix)
    - JOURS (**numjour**, nomjour, datevaleur, …)
    - CLIENT (**numclient**, nom, prenom, refadresse, codepostal,nb_enfant...)
    - VENDEURS (**numvendeur**,nom, prenom, adresse, codepostal, nb_enfants)
    - PRODUIT(**numproduit**, appelation, durée, numcatalogue, categorie, refrisque)

- Rappel SQL
  - **Langage Définition de Données LDD**
    - CREATE, DROP, ALTER
  - **Langage Manipulation de données LMD**
    - SELECT, INSERT, UPDATE, DELETE
    - clause WHERE prédicat :

      =, <>, <, >, <=, >=, LIKE %toto%, BETWEEN val1 AND val2

      IS NULL, IN, EXIST, ALL, ANY

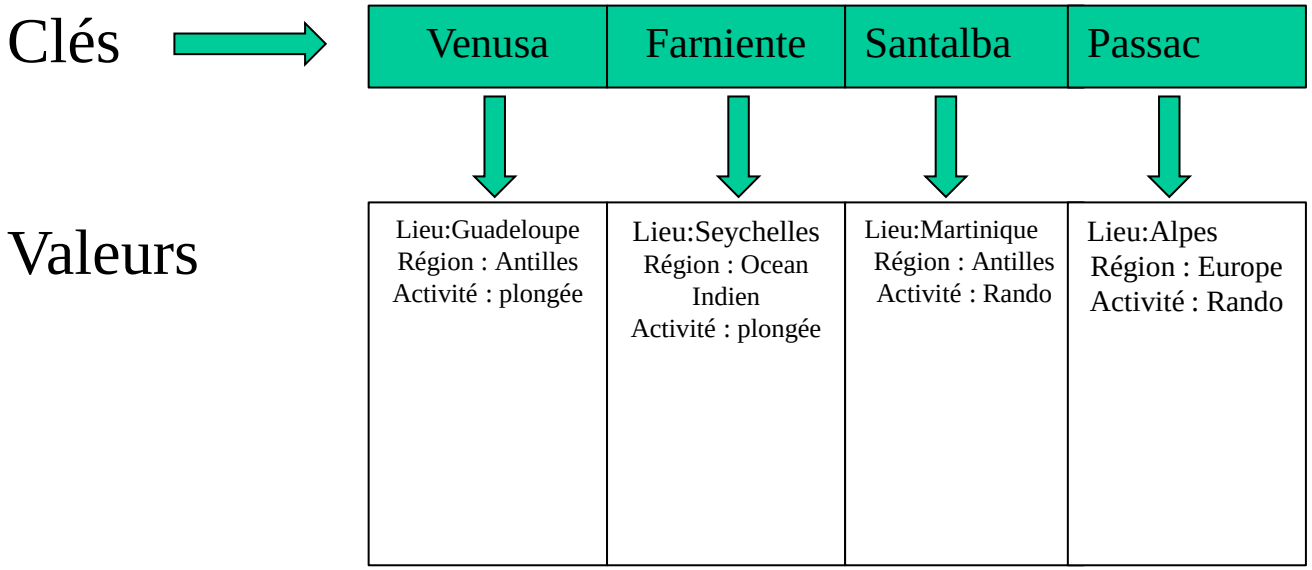
      GROUP BY, ORDER BY, HAVING

      OR, AND, NOT
  - **Langage Contrôle de Données LCD**
    - Commit, Rollback, Grant, Revoke

NOSQL

- 4 grandes familles modèles
  - Clés-Valeurs
  - Colonnes
  - Documents
  - Graphes

- La famille de base **Clés-Valeurs** (Key/value)
  - Stockage **orienté Clé**
  - énorme HashMap - table de hachage distribuée
  - 4 Opérations possibles dites CRUD
    - » Create (clé,valeur)
    - » Read (clé)
    - » Update (clé,valeur)
    - » Delete (clé)



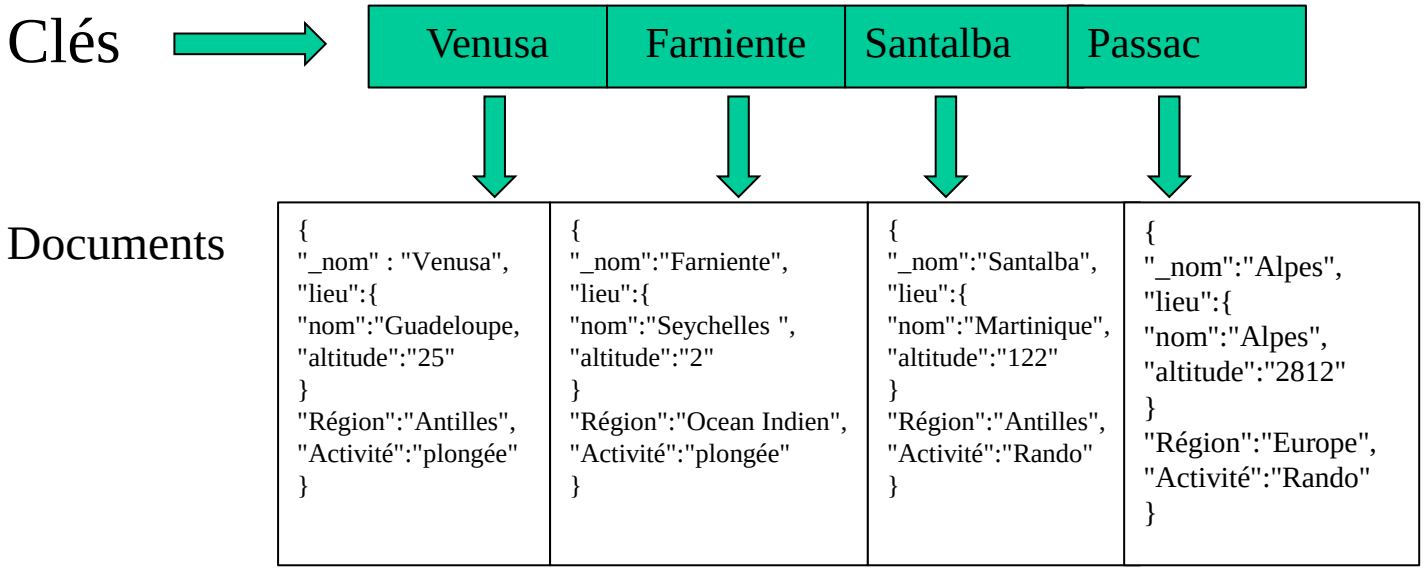
- La famille de base **Colonne** (Google Cloud BigTable, Hbase Apache ElasticSearch, SparkSQL)
  - Stockage **orienté Colonne**
  - Tables éclatées par attribut et identifiant et distribuées !!
  - Opérations d'agrégations et de corrélations entre tables
  - Exemple : Les activités en Guadeloupe (agrégation sur la clé **nom**)

Nom	Lieu
Venusa	Guadeloupe
Farniente	Seychelles
Santalba	Martinique
Passac	Alpe

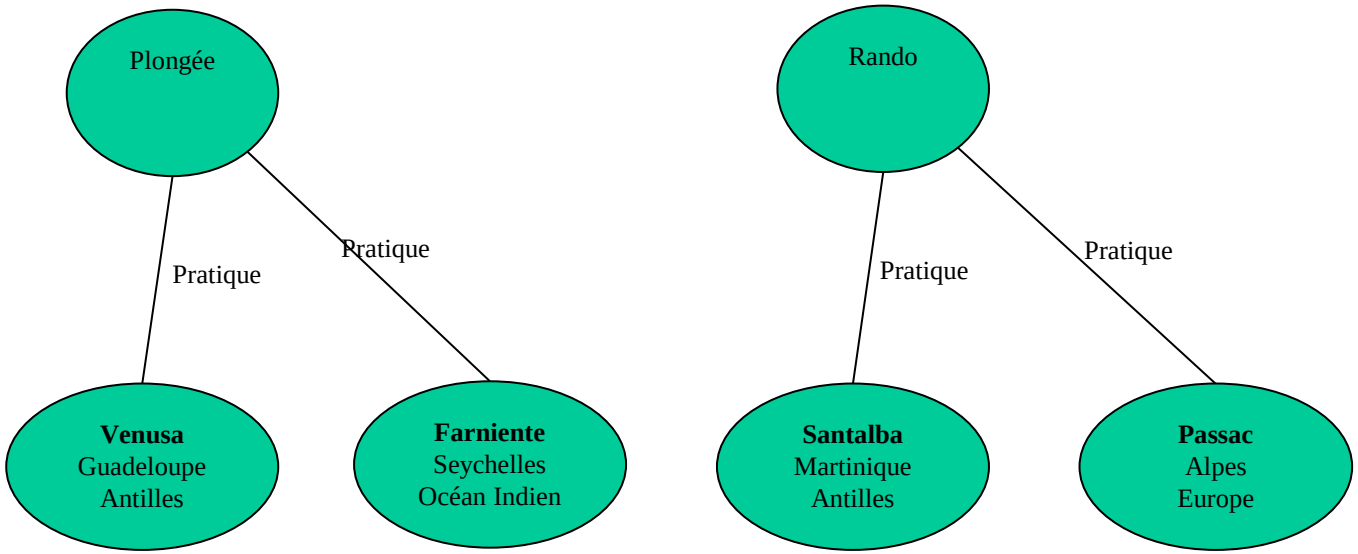
Nom	Region
Venusa	Antilles
Farniente	Océan Indien
Santalba	Antilles
Passac	Europê

Nom	Activité
Venusa	Plongée
Venusa	E-sport
Farniente	Plongée
Farniente	Char à voile
Santalba	Rando
Santalba	Hamac
Passac	Rando
Passac	Petanque

- La famille de base **Orientée Document** (MongoDB, Cassandra pour facebook, DynamoDB pour Amazon)
  - Stockage **orienté clés/ sous clés dans le documents**
  - **Langage Riche (documents)**
  - **Exemple : nom des lieux (lieu.nom) situés à plus de 100 m d'altitude (lieu.altitude>100) et pratiquant l'activité rando (activité= "Rando")**



- La famille de base **Orientée Graphe** (neo4j, flockdb twitter, MS Azure Cosmos DB)
  - Stockage **orienté graphes**
  - Réseaux, scoring d'information :



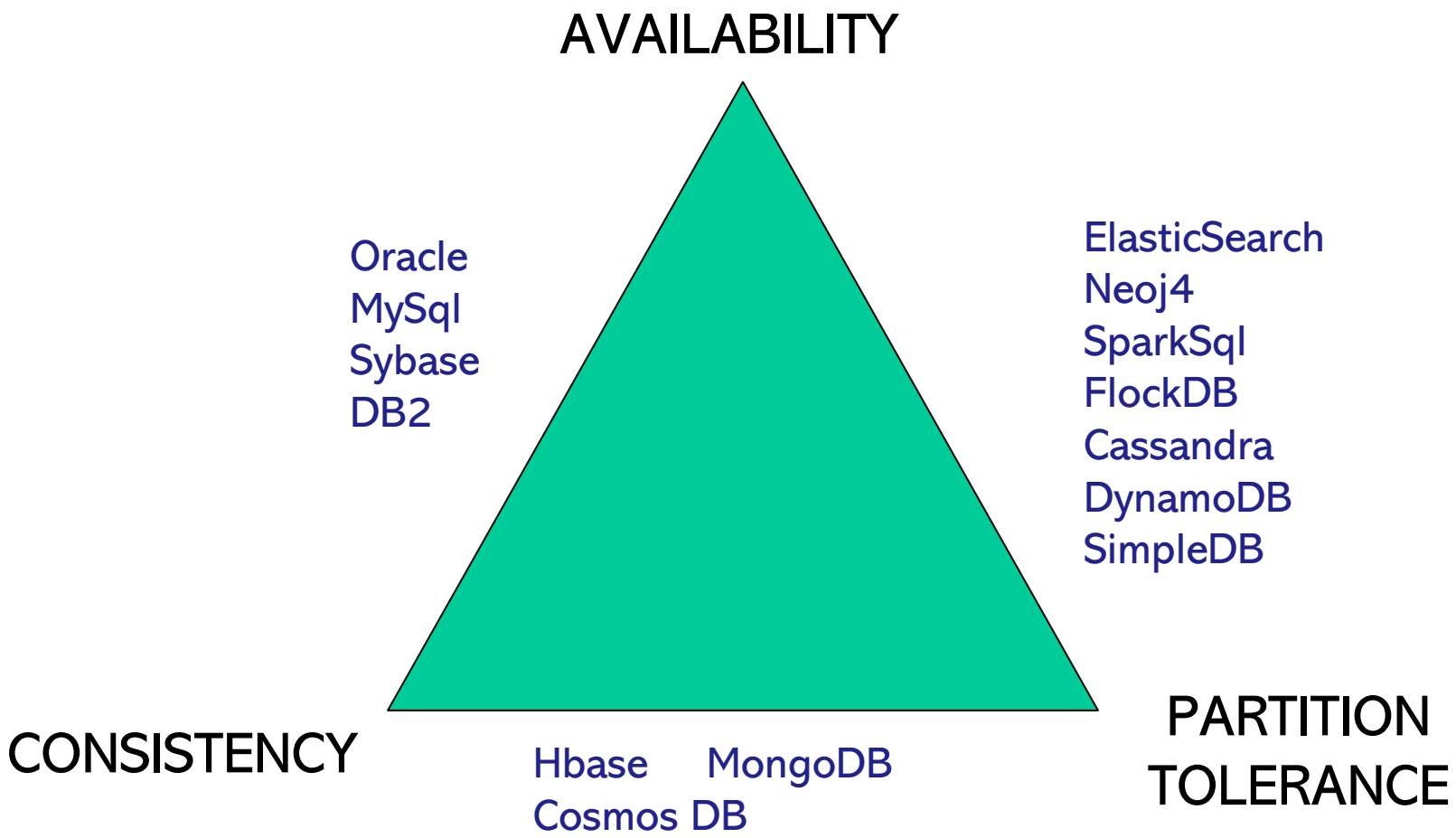
Base Relationnelle : ACID !

A*tomicité* : Une transaction doit être intègre : tout ou rien
C*ohérence* : Le contenu d'une base doit être cohérent avant et après la transaction
I*solation* : Les modifications ne sont visibles qu'après validation
D*urabilité* : Une fois la transaction validée, l'état de la base est permanent (reprise, panne, etc…)

Base NoSQL

C*onsistency* (Cohérence) : Une donnée n'a qu'un état visible quel que soit le nombre de réplicas
A*vailability* (Disponibilité) : Tant que le système est actif, la donnée est disponible
P*artition Tolerance* (Distribution) : Quel que soit la distribution, toute requête fournit un résultat correct

Le triangle de CAP (appelé Brewser en 2000)



En conclusion

- Evolution des données
  - **Grandes plateformes** autour du Web, moteurs de recherche, base de documents type cloud, réseaux sociaux,…
  - **Volume** considérable lié au données du web (videos,medias..) et aux capteurs (outils connectés, rapport, surveillance réseau,...)
  - **Données évolutives** sans structure connue à l 'avance.
- Evolution des besoins en gestion de données
  - répartir les données pour tenir les performances malgré la masse de données.
  - Nécessité de gérer des données hétérogènes et évolutives
- Constat
  - SGBD Relationnelles traditionnelles ne sont pas adaptés en **performances, évolutivité, flexibilité pour le traitement de données évolutives à grand volume dans un contexte distribué**.
  - Maturité du NoSql,
  - Emergence du NewSQL depuis 2021: issu des SGBDR mais dédié au traitement transactionnel en ligne, cherche à obtenir les propriétés des systèmes NoSql tout en maintenant les propriétés ACID