Découverte des bases de données

C. BENSARI

Plan

- Pourquoi une Base De Données (BDD) ?
- Définition d'une BDD
- Système de Gestion de Base de Données
- Conception d'une BDD
- Langage de manipulation de données (SQL)

Cas réel

- Un store de vente de matériel informatique:
  - Gestion des commandes
  - Gestion des livraisons
  - Gestion du stock
  - Gestion des clients
  - Gestion des fournisseurs
  - Gestion des compagnes promotionnelles
  - Gestion des conventions externes

➔ Comment optimiser le rendement du magasin en prenant en compte tous les aspects de gestion ?

Définition d'une base données

- Est un ensemble de données structurées de manière à pouvoir les récupérer (consultation), gérer (créer, modifier, supprimer) et mettre à jour
- Elle sont stockées dans des fichiers sur un appareil de stockage
- Peuvent être consultées ou modifiées par un Système de Gestion de Bases de Données

Un peu d'histoire

- 1960: BDD hiérarchiques (ex. IMS de IBM)
- 1970: BDD réseau (ex. IDS.II de BULL)
- 1970: BDD relationnelles (ex. MySQL de Oracle)
- 1980: BDD orientées objets (ex, O2)
- 2000: BDD NOSQL (ex. MongoDB)

Système d'information (SI)

- Un ensemble de moyens (matériels, logiciels, ..) mis en œuvre afin de gérer l'information
- Un SI possède 4 fonctions essentielles :
  - Collecte de l'information
  - Stockage de l'information
  - Traitement de l'information
  - Diffusion de l'information
- Un SGBD est mis au service d'un SI pour accomplir ses fonctions

Système de Gestion de Base de Données

- Un SGBD est un outil permettant de manipuler les donnée efficacement (CRUD: Création, Recherche, Update, Delete)

- Possède un système multi utilisateurs: gestion d'accès et de confidentialité

- Possède un système de monitoring afin d'analyser les transactions sur la BDD

Conception d'une base de données

- MERISE: méthode d'analyse et de conception des SI basée sur le principe de la séparation des données et des traitements

- La représentation graphique et structurée des informations stockées par un SI est appelée « Modèle Conceptuel de Données » (MCD)

- L'élaboration d'un MCD passe par les étapes suivantes :

  - Définition des règles de gestion

  - Définition du dictionnaire des données

  - Détermination des dépendances fonctionnelles entre les données

  - La mise en place du MCD

Conception d'un MCD
Définition des règles de gestion

- Définir les besoins des futurs utilisateurs de l'application
  - Exemples de règles de gestion:

    Pour chaque article, on doit connaitre son nom, sa référence, sa marque, son fournisseur, son prix, ..

    Pour chaque commande, on doit connaitre le nom du client, le nombre d'article, le montant total de la commande, ..

- Les règles de gestion sont souvent fournies par la MOA (Maitrise d'ouvrage) du projet

Conception d'un MCD

Définition du dictionnaire des donnée

- Définir un document qui regroupe toutes les données définies lors de la rédaction des règles de gestion. C'est données figureront dans le MCD

- Exemple de dictionnaire pour une donnée :

Conception d'un MCD

Définition du dictionnaire des données

Code Mnémonique	Désignation	Type	Taille	Remarque
id_article	Identifiant d'un article	N	10	
nom_article	Nom de l'article	A	30	
Marque_article	Marque de l'article	A	30	
prix_vente_article	Prix de vente de l'article	N	4	
Date_achat_article	Date d'achat	Date	10	Au format JJ/MM/AAAA
id_commande	Identifiant de la commande	N	10	
date_commande	Date de la commande du client	Date	10	Au format JJ/MM/AAAA
id_client	Identifiant d'un client	N	10	
nom_client	Nom d'un client	A	30	
prenom_client	Prénom d'un client	A	30	

Conception d'un MCD

Définition des dépendances fonctionnelles

- Une dépendance fonctionnelle est présente entre deux données P1 et P2 si et seulement si une occurrence (valeur) de P1 permet de connaître une et une seule occurrence de P2

- À partir du dictionnaire de données on peut trouver les dépendances suivantes :

  - id_article ---> nom_article, marque_article, prix_vente_article, date_achat_article
  - id_commande -- > date_commande, id_article, id_client
  - id_client -- -> nom_client, prenom_client

Conception d'un MCD

Le Modèle de Conceptuel de Données

- Les entités

Article
id_article
nom_article
marque_article
….

Commande
id_commande
date_commande
Id_client
….

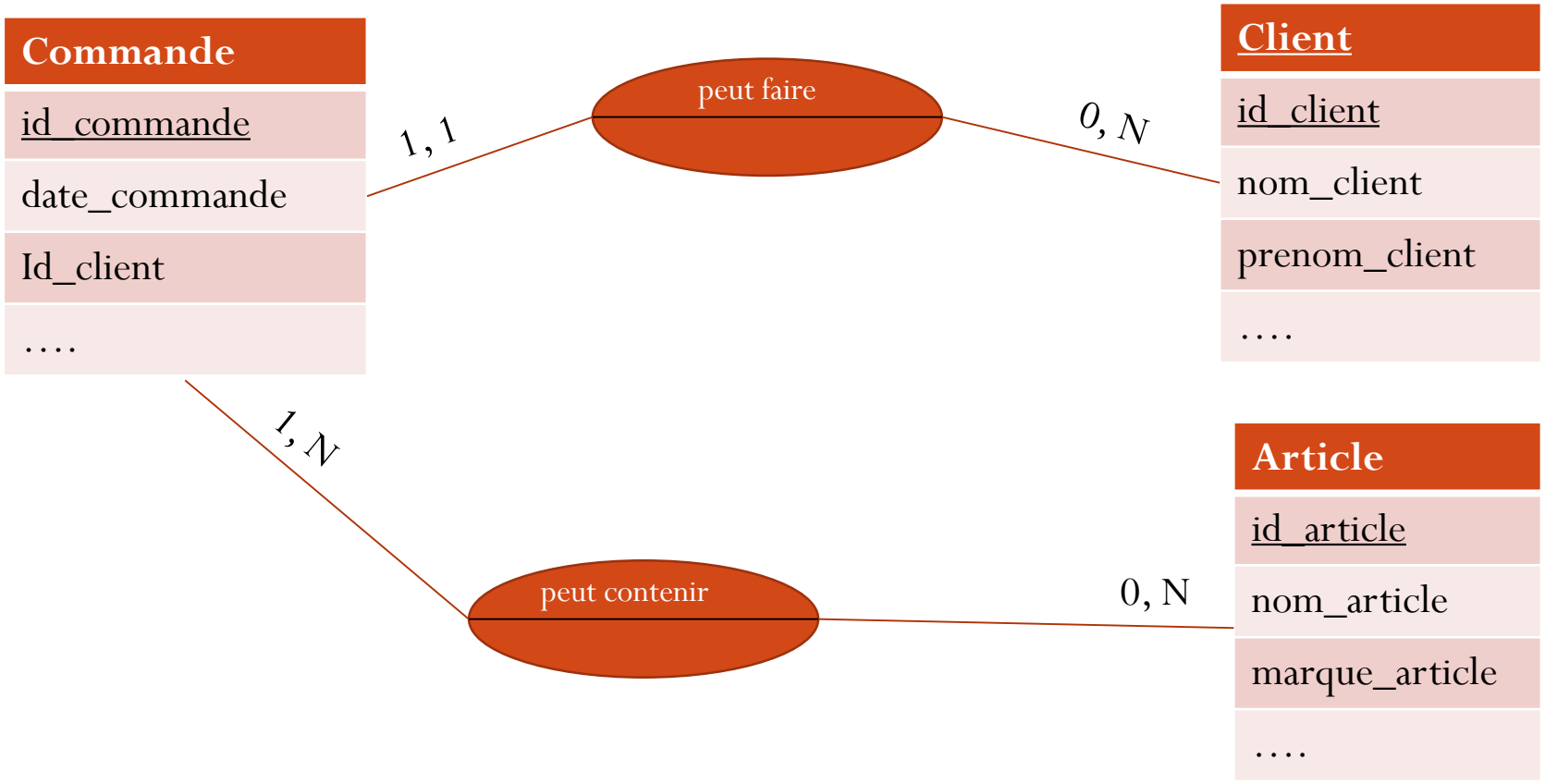
Client
id_client
nom_client
prenom_client
….

id_client	nom_client	prenom_client
1	DUPONT	David
2	LEGRAND	Stéphan
3	LECLAIR	Hugo

Conception d'un MCD

Le Modèle de Conceptuel de Données

- Les associations



Conception d'un Modèle Logique de Données (MLD)

- Un MLD est une représentation logique d'un MCD
- Trois règles à suivre pour réussir la conception d'un MLD :
  - Règle 1:Une entité est transformée en une table. La table doit contenir une clé primaire ainsi que les colonnes
  - Règle 2 : Une association de cardinalité « 1,N » est traduite par une clé étrangère dans la table du coté de la cardinalité 1
  - Règle 3 :Une association de cardinalité « N,N » est traduite par une table qui a comme clé primaire composée des deux clés étrangères référençant des deux tables

Exercices

- Exercice1 : Commandes

Dans cet exercice, vous allez modéliser le SI d'une société. Cette dernière passe des commandes à des fournisseurs. Les informations sur les fournisseurs sont : leur numéro, nom, prénom, adresse et leur téléphone. Ces commandes sont datées et portent des produits (dont on connaît le numéro, la désignation, la couleur et le poids) vendus à un certain prix et selon une certaine quantité.

1. Identifiez les entités

2. Donnez le dictionnaire de données

3. Proposez trois variantes de MCD

4. Définissez le MLD correspondant à chaque variante

Langage de manipulation de données (SQL)

Langage de manipulation d'une base de données (SQL)

- SQL (Structured Query langage)
- Permet de communiquer avec une base de données
- Très utilisé pour les applications Web (pages dynamiques)
- Les instructions SQL sont regroupées en plusieurs catégories:
  - Langage de Définition des Données (LDD) : CREATE, ALTER, DROP
  - Langage d'Interrogation de Données (LID) : SELECT …
  - Langage de Manipulation de Données (LMD) : UPDATE, DELETE, INSERT
  - Langage de Contrôle de Données (LCD) : GRANT, REVOKE

Langage de définition de données LDD

- **Créer une base de données -CREATE DATABASE- : syntaxe générale**

CREATE DATABASE nom_bdd **IF NOT EXIST**

Langage de définition de données LDD

- **Créer une table -CREATE TABLE- : syntaxe générale**

CREATE TABLE nom_bdd.nom-table
{ (nom-col type-col [**DEFAULT** valeur]
[[**CONSTRAINT**] contrainte-col])*
[[**CONSTRAINT**] contrainte-table]*
|
AS requête-SQL
};

Légende :
{a | b} : a ou b
[option]
* : applicable autant de fois que souhaité
mot en capitale : mot clé

Langage de Définition de Données

- **Types de colonnes les plus connus :**
  - **INTEGER :** permet de stocker des entiers signés codés sur 4 octets.
  - **BIGINT :** ce type permet de stocker des entiers signés codés sur 8 octets.
  - **REAL :** réels comportant 6 chiffres significatifs codés sur 4 octets.
  - **DOUBLE PRECISION** : Ce type permet de stocker des réels comportant 15 chiffres significatifs codés sur 8 octets.
  - **NUMERIC[(précision, [longueur])]**
  - **CHAR(longueur) :** chaînes de caractères de longueur fixe
  - **VARCHAR(longueur)**
  - **DATE : date.**
  - **TIMESTAMP :** date et d'une heure.
  - **BOOLEAN :** valeurs Booléenne.
  - **MONEY :** valeurs monétaires.
  - **TEXT :** chaînes de caractères de longueur variable.

Langage de Définition de Données

- **contrainte-col**: contrainte sur une colonne
  - **NOT NULL** : ne prendre pas la valeur null
  - **PRIMARY KEY** : clé primaire
  - **UNIQUE :** la valeur de la colonne doit être unique sur l'ensemble de n-uplets
  - **REFERENCES nom-table [(nom-col)] [ON DELETE CASCADE] :** utililsée pour déclarer des clés étrangères dans la table
  - **CHECK (condition) :** vérifie lors de l'insertion si la valeur respecte la condition

Langage de Définition de Données

- **contrainte-table** : contraintes sur une table
  - **PRIMARY KEY (nom-col*)**
  - **UNIQUE (nom-col*)**
  - **FOREIGN KEY (nom-col*) REFERENCES nom-table [(nom-col*)] [ON DELETE CASCADE | SET NULL]:** utililsée pour déclarer des clés étrangères
  - **CHECK (condition)**

Exemple de création de table

- **CREATE TABLE** Doctorant
  (nom **VARCHAR(20),**
  prénom **VARCHAR(15)**,
  année_insc **DECIMAL(4) DEFAULT** 2003) ;

- **CREATE TABLE** Doctorant
  AS SELECT nom, prénom, année_inscr
  FROM Etudiant **WHERE** statut='Doctorant' ;

Exemple de PRIMARY KEY

- **CREATE TABLE** Pays
  (nom **VARCHAR(20) PRIMARY KEY** ,
  capitale **VARCHAR(20)** …)

- **CREATE TABLE** Employé
  (nom **VARCHAR(30)** ,
  prénom **VARCHAR(30)** ,
  age NUMBER **CHECK** (âge **BETWEEN** 16 **AND** 70)
  adresse **VARCHAR(60)** , …
  CONSTRAINT Pk_emp **PRIMARY KEY** (nom,
  prénom))

Exemple de FOREIGN KEY

- **CREATE TABLE** Etudiant (N°E …)
- **CREATE TABLE** Cours (NomCours …)
- **CREATE TABLE** Suit

  (N°Etud **CHAR(9)** ,

  NomC **VARCHAR(25)** ,

  PRIMARY KEY (N°Etud , NomC) ,

  FOREIGN KEY (N°Etud) **REFERENCES** Etudiant ,

  FOREIGN KEY (NomC) **REFERENCES** Cours)

Suppression d'une table

- **Supprimer une table**

DROP TABLE nom_table **[CASCADE CONSTRAINTS]**

CASCADE CONSTRAINTS

- Supprime toutes les contraintes de clé externe référençant cette table
- Si on cherche à détruire une table dont certains attributs sont référencés sans spécifier CASCADE CONSTRAINT→ refus

Modification d'une table

- **Modifier une table**

  ALTER TABLE nom-table

  { **RENAMETO** nouveau-nom-table |

  ADD ([(nom-col type-col [**DEFAULT** valeur]

  [contrainte-col])*] |

  MODIFY (nom-col [type-col] [**DEFAULT** valeur]

  [contrainte-col])* |

  DROP COLUMN nom-col [**CASCADE CONSTRAINTS**] |

  RENAME COLUMN old-name **TO** new-name

  }

Modification d'une table

- **Ajout ou modification de colonnes**

ALTER TABLE nom_table {**ADD | MODIFY**} ([nom_colonne type [contrainte], …])

- **Ajout d'une contrainte de table**

ALTER TABLE nom_table **ADD** [**CONSTRAINT** nom_contrainte] contrainte

NB: Si des données sont déjà présentes dans la table au moment où la contrainte d'intégrité est ajoutée, toutes les lignes doivent vérifier la contrainte. Dans le cas contraire, la contrainte n'est pas posée sur la table.

- **Renommer une colonne**

ALTER TABLE nom_table **RENAME COLUMN** ancien_nom **TO** nouveau_nom

- **Renommer une table**

ALTER TABLE nom_table **RENAME TO** nouveau_nom

Langage de manipulation de données (LMD)

Insertion de n-uplets : INSERT INTO

- **Cas 1**: insérer des nouvelles donnees en spécifiant les valeurs

INSERT INTO nom_table(nom_col_1, nom_col_2, …) VALUES (val_1, val_2, …)

- **Cas 2:** Données provenant d'une autre table. La syntaxe est la suivante :

INSERT INTO nom_table (nom_col1, nom_col2, …) **SELECT** …

Remarque: Le SELECT peut contenir n'importe quelle clause sauf un **ORDER BY** et dont le résultat a le même schéma que nom-table

Exp: INSERT INTO mairie(lieu, region) SELECT lieu, region FROM ville

Modification de données d'une table

- **Modification de n-uplets**

- **UPDATE** nom_table

SET nom_col_1 = {expression_1 | (SELECT …) },

nom_col_2 = {expression_2 | (SELECT …) },

…

nom_col_n = {expression_n | (SELECT …) }

WHERE predicat

Suppression de données d'une table

- **Suppression de n-uplets**

DELETE FROM nom_table **WHERE** predicat

- Remarque:

En l'absence de clause WHERE, toutes les lignes de la table seront supprimées

Langage de requêtes: Interroger une base

- **Syntaxe générale de la commande SELECT**

SELECT [DISTINCT] {* | liste_express}
[FROM liste_tables]
[WHERE condition [ALL]]
[GROUP BY liste_expression]
[HAVING condition]
[ORDER BY liste expression]

Syntaxe générale d'une requête SELECT

- SELECT * : permet de choisir toutes les colonnes d'une table
- SELECT col1, col2 : permet de choisir col1, col2 d'une table
- SELECT distinct : permet de choisir des valeurs distinctes d'une table
  - Exemple : Les différents prénoms des employés

Exemple d'utilisation de DISTINCT

- SELECT DISTINCT prenom, nom FROM employes

NB: DISTINCT permet de ne retenir qu'une occurrence de n-uplet dans le cas où une requête produit plusieurs n-uplets identiques

Exercices : Select simple

- 1 : Sélectionner toutes les colonnes de la table SERV.
- 2 : Sélectionner d'une autre manière ces colonnes.
- 3 : Sélectionner les colonnes SERVICE et NOSERV de la table SERV.
- 4 : Sélectionner toutes les colonnes de la table EMP.
- 5 : Sélectionner les emplois de la table EMP.
- 6 : Sélectionner les différents emplois de la table EMP.

Exemple d'utilisation du SELECT

- **Sélection**
  - SELECT * FROM relation WHERE prédicat

- **Produit cartésien**
  - SELECT * FROM relation_1, relation_2

- **équi-jointure**
  - SELECT * FROM relation_1, relation_2 WHERE relation_1.cle_primaire = relation_2.cle_etrangere

Exercices : Selects simples

- 07 : Sélectionner les employés du service N°3.
- 08 : Sélectionner les noms, prénoms, numéro d'employé, numéro de service de tous les techniciens.
- 09 : Sélectionner les noms, numéros de service de tous les services dont le numéro est supérieur à 2.
- 10 : Sélectionner les noms, numéros de service de tous les services dont le numéro est inférieur ou égal à 2.
- 11: Sélectionner les employés dont la commission est inférieure au salaire.

Select avec des valeurs Nulles

- Une valeur nulle est différente de la valeur zéro. Elle correspond à une valeur non renseignée
- Il faut comprendre cette valeur comme l'ensemble vide et imaginer les calculs en découlant :
  - NULL + 3 donne NULL.
  - NULL = 0 est faux.
  - NULL n'est pas inférieur à 3 et NULL n'est pas supérieur à 3.
- Pour détecter une clause NULL, il faut utiliser le prédicat IS NULL. L'inverse étant IS NOT NULL
- Pour transformer la valeur NULL à une autre valeur, on utilise la fonction IFNULL (champ, new_value)
  - IFNULL(champ, 0) donne la valeur 0 si le champ est null sinon donne sa valeur.
  - Important pour protéger les colonnes ne pouvant pas être nulles.

Exercices : Travailler avec NULL

- 12 : Sélectionner les employés qui ne touchent jamais de commission.
- 13 : Sélectionner les employés qui touchent éventuellement une commission et dans l'ordre croissant des commissions.
- 14 : Sélectionner les employés qui ont un chef.
- 15 : Sélectionner les employés qui n'ont pas de chef.

SELECT : Les opérateur logiques AND, OR et NOT

- Rappels logiques
  - VRAI AND VRAI donne VRAI
  - VRAI AND FAUX donne FAUX
  - FAUX AND FAUX donne FAUX
  - VRAI OR VRAI donne VRAI
  - VRAI OR FAUX donne VRAI
  - FAUX OR FAUX donne FAUX
- AND prioritaire sur le OR : il faut donc utiliser des parenthèses pour le OR
- NOT inverse le résultat logique de l'opération

Select avec prédicats AND, OR et NOT

- 16 : Sélectionner les noms, emploi, salaire, numéro de service de tous les employés du service 5 qui gagnent plus de 20000 €.
- 17 : Sélectionner les vendeurs du service 6 qui ont un **revenu** mensuel supérieur ou égal à 9500 €.
- 18 : Sélectionner dans les employés tous les présidents et directeurs.
- 19 : Sélectionner les directeurs qui ne sont pas dans le service 3.
- 20 : Sélectionner les directeurs et « les techniciens du service 1 ».
- 21 : Sélectionner les « directeurs et les techniciens » du service 1.
- 22 : Sélectionner les employés du service 1 qui sont directeurs ou techniciens.
- 23 : Sélectionner les employés qui ne sont ni directeur, ni technicien et travaillant dans le service 1.

Les opérateurs : LIKE, IN et BETWEEN

- **LIKE**
  - Syntaxe : expression **LIKE** chaîne_modèle
  - chaîne_modèle est une chaîne de caractères pouvant contenir l'un des caractères jokers suivants:
    - '_' underscore remplace 1 caractère exactement.
    - '%' remplace une chaîne de caractères de longueur quelconque, y compris de longueur nulle.
- **IN**
  - expression_1 **IN** (expression_2, expression_3, …)
  - Vrai si expression_1 est égale à une des expressions de la liste entre parenthèses.
  - **NOT IN** pour inverser cette condition
- **BETWEEN**
  - expr1 **BETWEEN** expr2 **AND** expr3
  - Vrai si expr1 est comprise entre expr2 et expr3, bornes incluses.

Exercices: IN, BETWEEN, LIKE

- 24 : Sélectionner les employés qui sont techniciens, comptables ou vendeurs.
- 25 : Sélectionner les employés qui ne sont ni technicien, ni comptable, ni vendeur.
- 26 : Sélectionner les directeurs des services 2, 4 et 5.
- 27 : Sélectionner les subalternes qui ne sont pas dans les services 1, 3, 5.
- 28 : Sélectionner les employés qui gagnent entre 20000 et 40000 euros, bornes comprises.
- 29 : Sélectionner les employés qui gagnent moins de 20000 et plus de 40000 euros.
- 30 : Sélectionner les employés qui ne sont pas directeur et qui ont été embauchés en 88.
- 31 : Sélectionner les directeurs des services 2 ,3 , 4, 5 sans le IN.
- 32 :Sélectionner les employés dont le nom commence par la lettre M.
- 33 : Sélectionner les employés dont le nom se termine par T.
- 34 : Sélectionner les employés ayant au moins deux E dans leur nom.
- 35 : Sélectionner les employés ayant exactement un E dans leur nom.
- 36 : Sélectionner les employés ayant au moins un N et un O dans leur nom.
- 37 : Sélectionner les employés dont le nom s'écrit avec 6 caractères et qui se termine par N.
- 38 : Sélectionner les employés dont la troisième lettre du nom est un R.
- 39 : Sélectionner les employés dont le nom ne s'écrit pas avec 5 caractères.

Select : ORDER By

- Les critères de tri sont spécifiés dans une clause ORDER BY figurant après la clause FROM et la clause WHERE
- La syntaxe est la suivante :
  - ORDER BY nom_col1 | num_col1 [DESC], [nom_col2 | num_col2 [DESC], ….
- Le tri se fait selon la colonne 1, les lignes ayant la même valeur dans la colonne 1 sont triées selon la colonne 2, etc…
- Pour chaque colonne le tri peut être ascendant (ASC option par défaut) ou descendant (option DESC)
- Pour simplifier, il n'est pas indispensable de nommer explicitement la colonne mais en donnant sa position dans le Select.

Exercices : ORDER BY

- 40 : Trier les employés (nom, prénom, n° de service, salaire) du service 3 par ordre de salaire croissant.
- 41 : Trier les employés (nom, prénom, n° de service , salaire) du service 3 par ordre de salaire décroissant.
- 42 : Idem en indiquant le numéro de colonne à la place du nom colonne.
- 43 : Trier les employés (nom, prénom, n° de service, salaire, emploi) par emploi, et pour chaque emploi par ordre décroissant de salaire.
- 44 : Idem en indiquant les numéros de colonnes.
- 45 : Trier les employés (nom, prénom, n° de service, commission) du service 3 par ordre croissant de commission.
- 46 : Trier les employés (nom, prénom, n° de service, commission) du service 3 par ordre décroissant de commission, en considérant que celui dont la commission est nulle ne touche pas de commission.

OPERATEURS ENSEMBLISTES : UNION, INTERSECTION, ..

- **INTERSECT**
  - Elle donne pour résultat les lignes communes entre la première et la seconde requête
- **UNION**
  - Elle donne pour résultat les lignes de la première requête suivies de celles de la deuxième requête **sans** les doublons.
- **UNION ALL**
  - Elle donne pour résultat les lignes de la première requête suivies de celles de la deuxième requête **avec** les doublons.
- **MINUS (Oracle)**
  - Elle donne pour résultat les lignes de l'union des deux requêtes qui ne sont pas dans la deuxième requête.

OPERATEURS ENSEMBLISTES: LA JOINTURE

- C'est une opération qui permet d'associer plusieurs tables dans une requête et d'obtenir des résultats avec des informations provenant de plusieurs tables.
- SELECT … FROM table1, table2, table3, …., tableN
- Dans le cas de deux tables ayant une colonne en commun, on sélectionne les couples ayant une valeur commune dans cette colonne, on dit alors que l'on réalise une JOINTURE des deux tables. bien entendu cette sélection est exprimée à l'aide d'une clause WHERE.
- Lorsqu'il y a ambiguïté sur la table à laquelle appartient une colonne, il faut préfixer cette colonne avec le nom de la table (ou son **alias**).
- Exemple: afficher la liste des employés avec la ville dans laquelle ils travaillent.
- Pour une jointure d'une table avec elle-même, il faut utiliser des alias différents pour chaque table.

Type de jointures

- **INNER JOIN** : une jointure interne, elle permet de relier deux tables qui possèdent une colonne commune et que la condition d'égalité des valeurs de la colonne commune est vrai. C'est le type de jointure le plus utilisé
  - **Syntaxe** :

    SELECT * **FROM** table_1 **AS** A **INNER JOIN** table_2 **AS** B

    ON A.id = B.id
- **CROSS JOIN :** une jointure croisée, permet de lier chaque ligne de la première table avec chaque ligne de la deuxième table.
  - **Syntaxe :**

    SELECT * **from** table_1 **CROSS JOIN** table_2

Type de jointures

- **LEFT JOIN (LEFT OUTER JOIN)** : jointure externe, permet d'avoir toutes les lignes de la table de gauche même si il y a absence d'égalité sur la colonne commune
  - Syntaxe :

    SELECT * FROM tab1 **AS** x **LEFT JOIN** tab_2 **AS** y

    ON x.id = y.id
- **RIGHT JOIN (RIGHT OUTER JOIN)** : jointure externe, permet d'avoir toutes les lignes de la table de droite même si il y a absence d'égalité sur la colonne commune
  - Syntaxe :

    SELECT * FROM tab1 **AS** x **RIGHT JOIN** tab_2 **AS y**

    ON a.id = y.id

Type de jointures

- **FULL JOIN** (FULL OUTER JOIN): jointure externe, permet de retourner les lignes de la table de gauche et de droite si l'une des condition est vraie (non disponible sur MYSQL)
  - Syntaxe :

    SELECT * **FROM** table_1 **AS** x **FULL JOIN** table_2 **AS** y

    ON x.id = y.id

    Equivalent à

    SELECT * **FROM** tab1 **AS** x **LEFT JOIN** tab_2 **AS** y

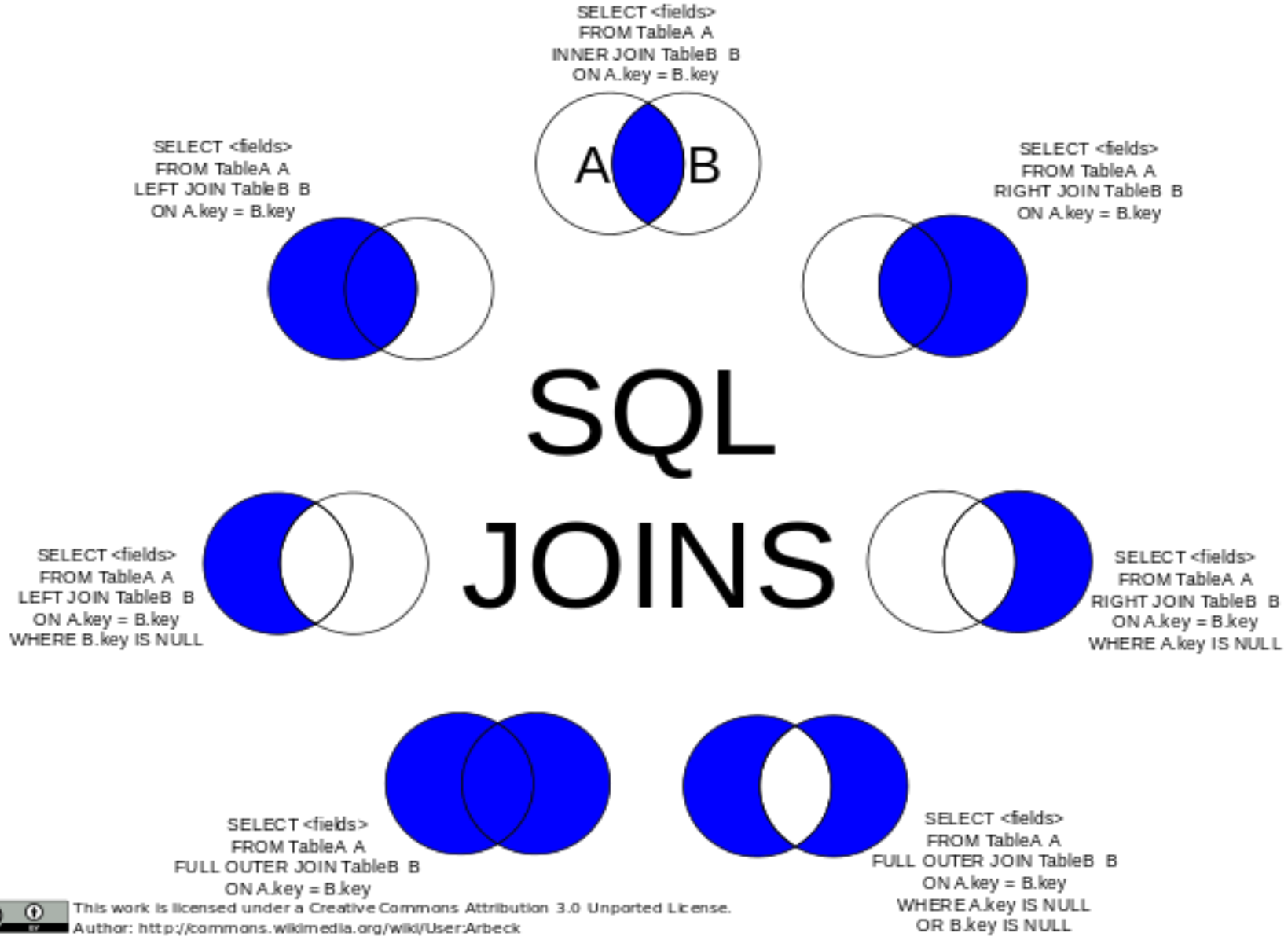
    ON x.id = y.id

    UNION

    SELECT * **FROM** tab1 **AS** x **RIGHT JOIN** tab_2 **AS** y

    ON x.id = y.id

Résumé



Exercices : Make a join !

- 47 : Sélectionner le nom, le prénom, l'emploi, le nom du service de l'employé pour tous les employés.
- 48 : Sélectionner le nom, l'emploi, le numéro de service, le nom du service pour tous les employés.
- 49 : Idem en utilisant des alias pour les noms de tables.
- 50 : Sélectionner le nom, l'emploi, suivis de toutes les colonnes de la table SERV pour tous les employés.
- 51 : Sélectionner les nom et date d'embauche des employés suivi des nom et date d'embauche de leur supérieur pour les employés plus ancien que leur supérieur, dans l'ordre nom employés, noms supérieurs.
- 52 : Sélectionner sans doublon les prénoms des directeurs et « les prénoms des techniciens du service 1 » avec un UNION.
- 53 : Sélectionner les numéros de services n'ayant pas d'employés sans une jointure externe
- 54 : Sélectionner les services ayant au moins un employé.