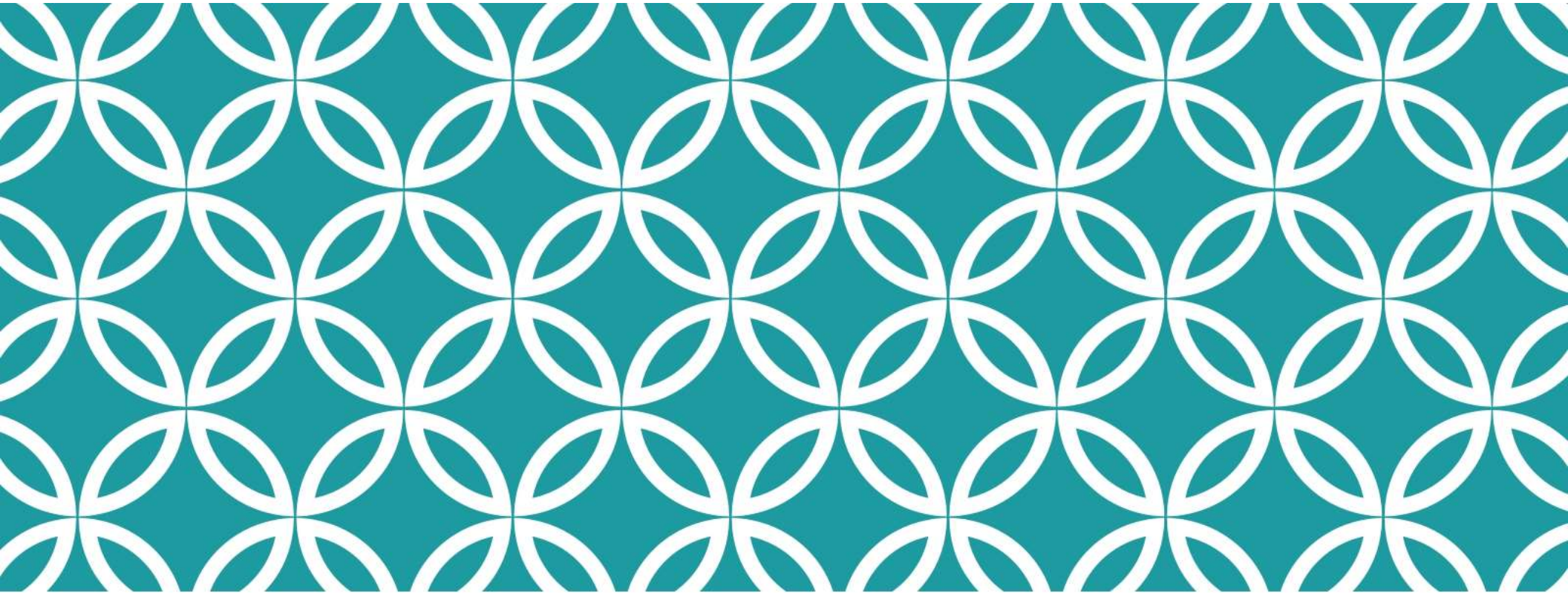


SGBDR

SOMMAIRE

SGBDR

Systèmes de Gestion de Bases de Données Relationnelles

Outil pour

- Structurer
- Stocker
- Interroger
- Garantir l'**intégrité** des données

Processus actif

Accessible via un port de communication spécifique

Utilisation du langage SQL pour interagir avec ce système

- Structured Query Language

SGBDR

DDL

- Data Definition Langage
- Définition de structure
- CREATE
- ALTER
- DROP
- TRUNCATE
- COMMENT
- RENAME

DML

- Data Manipulation Langage
- Manipulations
- SELECT
- INSERT
- UPDATE
- DELETE
- CALL
- LOCK TABLE

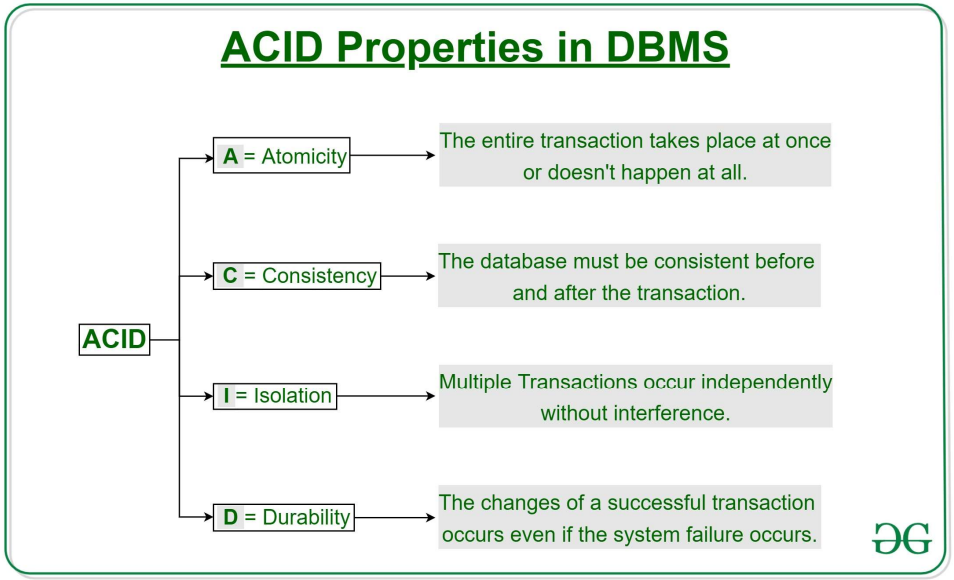
DCL

- Data Control Langage
- Sécurisation des données
- GRANT
- REVOKE

TCL

- Transaction Control Langage
- Gestion des transactions
- COMMIT
- ROLLBACK
- SAVEPOINT
- SET TRANSACTION

SGBDR



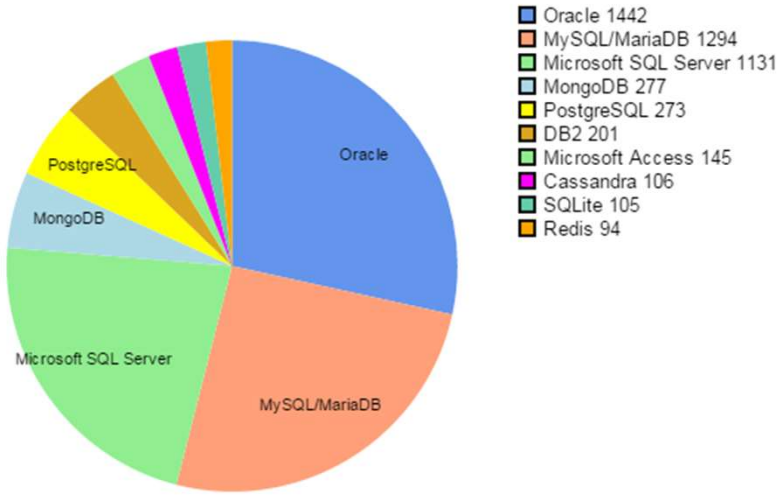
Transactions

- Suite d'instructions
- ACID
  - Atomique — Indivisible, tout ou rien
  - Cohérente — Le contenu final (dans la base de données) doit être cohérent
  - Isolée — Une transaction ne doit pas interférer avec une autre
  - Durable — Le résultat final est conservé indéfiniment (persistance de la donnée)

SGBDR

Quelques serveurs

- MySQL
- MariaDB
- Oracle
- PostgreSQL
- Microsoft SQL Server
- SQLite
- …



SGBDR

ORACLE

Oracle Database est un système de gestion de base de données relationnelle et relationnel-objet, crée par Oracle dans les années 70.
Elle est la première **database** conçue pour le grid computing. Le **grid computing** en entreprise est la technique la plus flexible et rentable pour gérer les systèmes informatiques et les applicatifs.

•Licence : Commercial
•Dernière version : 19c
•Ecrit en : Java, C et C++

AVANTAGES DE ORACLE
•Bonne capacité de sauvegarde et de récupération des données
•Régulièrement mis à jour
•Grande **portabilité**
•Gère facilement plusieurs bases de données au sein d'une même **transaction**
•La base de données la plus populaire selon le classement DB-Engines

INCONVÉNIENTS DE ORACLE
•Le prix
•Un système difficile à maîtriser

SGBDR

MICROSOFT SQL SERVER

Microsoft SQL Server, abrégé MSSQL est un **SGBD relationnel** crée par Microsoft en 1989. Cet outil se démarque de la concurrence grâce à un large choix d'options offertes selon la version choisie.
•Licence : Licence propriétaire et EULA
•Dernière version : 2019
•Ecrit en : C++, C et C#

AVANTAGES DE SQL SERVER
•Bonne **sécurité** des données
•Facile à installer et à configurer
•De nombreux outils pour gérer l'ensemble des tâches en entreprise

INCONVÉNIENTS DE SQL SERVER
•Le prix
•Le manque de **compatibilité** avec des produits ne provenant pas de Microsoft
•Besoin de machines performantes pour fonctionner correctement

SGBDR

MYSQL

MySQL est un **SGBD** (Système de Gestion de Base de Données) relationnelle, crée par MySQL AB en **1995**.
Appréciée des professionnels et des particuliers, elle est la **base de données la plus utilisée au monde**.

•Licence : Licence publique générale GNU version 2 et licence propriétaire
•Dernière version : 8.0 RC1
•Ecrit en : C et C++

AVANTAGES DE MYSQL
•La base de données la plus utilisée au monde
•Facile à utiliser
•De bonnes **performances**
•Plusieurs fonctionnalités pour sécuriser ses données
•Open-source

INCONVÉNIENTS DE MYSQL
•Difficilement **scalable**, les performances du système se détériorent à partir d'un certain volume de données

SGBDR

POSTGRESQL

PostgreSQL est un **SGBD relationnelle** et objet créé par le groupe PostgreSQL en 1996. Il s'agit d'un outil libre, non contrôlé par une entreprise, mais par une communauté mondiale de développeurs et d'organisations.
•Licence : Licence PostgreSQL
•Dernière version : 13.4
•Ecrit en : C

AVANTAGES DE POSTGRESQL
•**Open-source**
•Facile à utiliser
•Possède un type de données défini par l'utilisateur
•Une grande **communauté**

INCONVÉNIENTS DE POSTGRESQL
•L'un des plus mauvais en termes de performances
•La **réplication** est complexe
•Difficile à installer

SGBDR

Un SGBDR peut gérer plusieurs bases de données

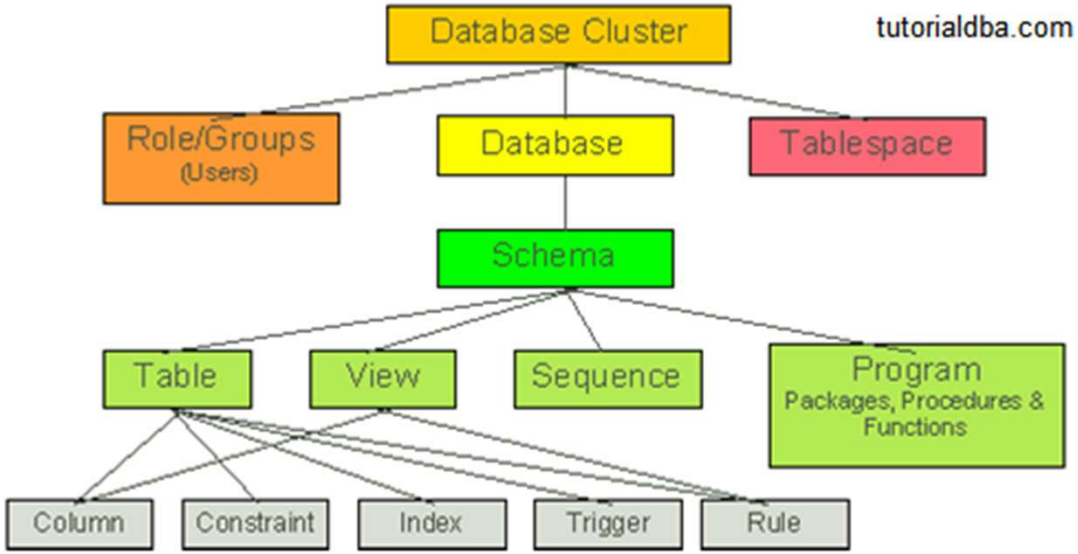
Une base de données peut contenir plusieurs tables

Une table possède plusieurs colonnes

Chaque enregistrement est identifié grâce à une clé primaire

On peut créer un lien entre enregistrements grâce à la clé étrangère

SGBDR



Modèle entité association	Base de données
Entité	Table
Propriété	champ
Identifiant	clé primaire
Occurrence	enregistrement

SGBDR

Dans la base de données, voici la table « client »

ID	NOM	PRENOM
1	PERROUAULT	Jérémy
2	PERROUAULT	Alissa
3	CESBRON	Martin

Les colonnes sont
- ID, NOM et PRENOM

Une clé primaire est un élément obligatoire
- Ici, la colonne « ID » est la clé primaire de la table

SGBDR

ID	NOM	PRENOM
1	PERROUAULT	Jérémy
2	PERROUAULT	Alissa
3	CESBRON	Martin

ID_CLIENT	ID_PRODUIT
1	1
1	2
2	1

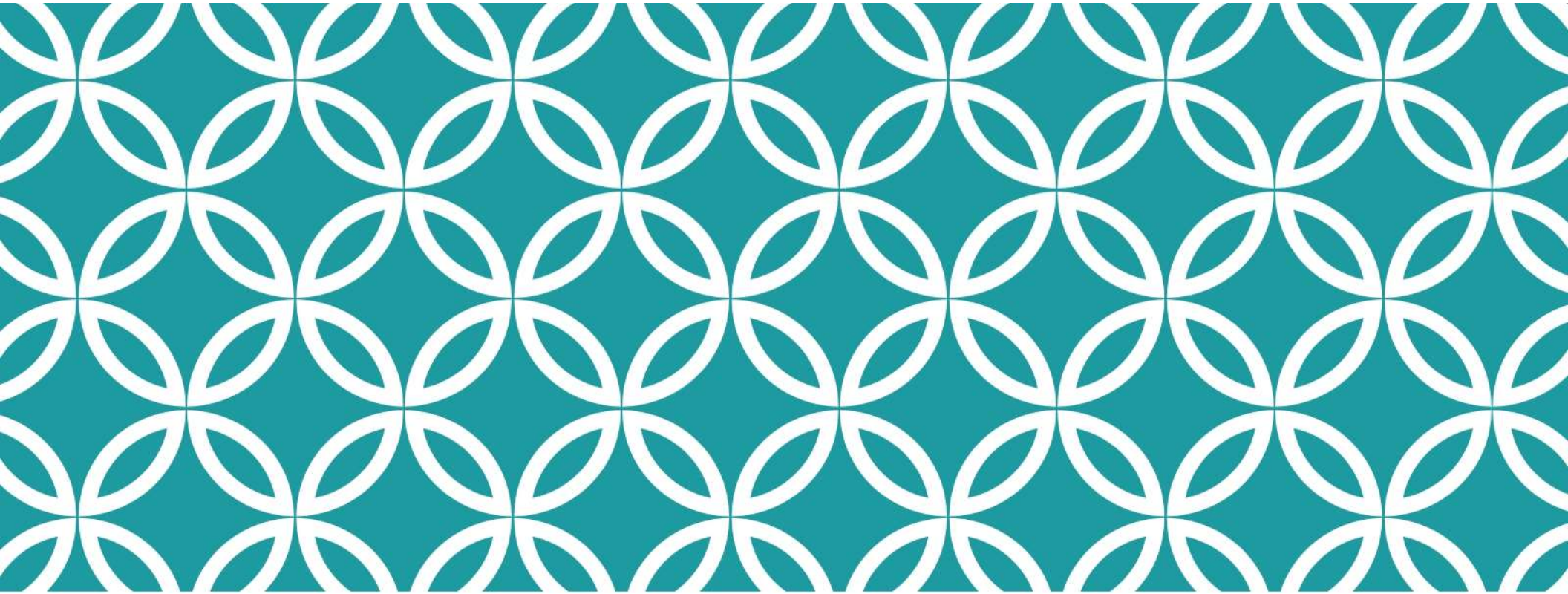
ID	LIBELLE	PRIX
1	GoPRO HERO 5	429.99
2	GoPRO KARMA	699.99

Dans cet exemple

Les « ID » sont des clés primaires

« ID_CLIENT » et « ID_PRODUIT » sont

- Des clés primaires
- Des clés étrangères



STRUCTURE DES DONNÉES

Langage DDL

STRUCTURE DES DONNÉES

Langage DDL (Data Definition Language – Data Description Language)

Langage de structuration des données

STRUCTURE DES DONNÉES

CREATE

- Création d'un élément de structure

ALTER

- Modification d'un élément de structure

DROP

- Suppression d'un élément de structure

STRUCTURE DES DONNÉES

CREATE DATABASE
- Créer une base de données

CREATE TABLE
- Créer une table

ALTER TABLE
- Modifier une table
  - Ajouter, modifier, supprimer une colonne
  - Ajouter, supprimer un index

DROP TABLE
- Supprimer une table

TRUNCATE TABLE
- Vider une table

STRUCTURE DES DONNÉES

Création d'une base de données (ou d'une table)

- Préciser la COLLATION par défaut (classement, système d'encodage des chaines)
- Système d'encodage
  - UTF-8
  - Latin-1
  - ASCII
- Comparaison de chaines
  - general_ci compare les valeurs sans prendre en compte la casse, ni les accents
  - bin compare les valeurs binaires (comparaison stricte)

Système de stockage UTF-8, comparaison sans accents ni casse

- utf8_general_ci

STRUCTURE DES DONNÉES

CREATE DATABASE

```
CREATE DATABASE nom_db COLLATE utf8_general_ci;
```

SHOW DATABASES

```
SHOW databases;
```

USE DATABASE

```
USE nom_db;
```

SHOW TABLES

```
SHOW tables;
```

STRUCTURE DES DONNÉES

CREATE TABLE

```
CREATE TABLE nom_db.matable (
  `nom colonne avec espace` TYPE OPTIONS,
  id INT NOT NULL,
  nom VARCHAR(100) NOT NULL,
  prenom VARCHAR(150) NOT NULL,
  age INT NOT NULL,
  id_parent INT NULL
) ENGINE = InnoDb;
```

Voir les propriétés d'un objet : ici, voir les champs de la table Desc matable

STRUCTURE DES DONNÉES

InnoDB	MyISAM
Default storage engine as of MySQL 5.5	Default storage engine before MySQL 5.5
ACID* compliant	Not ACID compliant
Transactional (Rollback, Commit)	Non-transactional
Row Level Locking	Table Level Locking
Row data stored in pages as per Primary Key order	No Particular order for data stored
Supports Foreign Keys	Does not support relationship constraint
No Full Text Search	Full Text Search

EXERCICE

Télécharger et installer *MySQL Server* *ou* *WAMP SERVER*

https://dev.mysql.com/downloads/installer/

https://dev.mysql.com/downloads/workbench/

https://www.wampserver.com/

Créer une base de données first_database

STRUCTURE DES DONNÉES

Types de données

Chaque champ est qualifié par un **type de données**. On choisi le type en fonction de l'information que l'on souhaite stocker.
Les types les plus courants sont :

char(taille), varchar(taille), text
Pour stocker du texte.

smallint, int, bigint
Pour stocker des entiers

real, numeric(lg, precision)
Pour stocker nombres à virgule

bool (ou boolean)
Pour stocker un valeur Vrai/Faux

Date, timestamp, time
Pour stocker des dates et heures

smallserial, serial, bigerial
Pour créer un compteur qui sera un entier

A retenir absolument

varchar()
int
real
bool
Date
timestamp

OPTIONS

NULL
NOT NULL
DEFAULT
PRIMARY KEY

STRUCTURE DES DONNÉES

ALTER TABLE

- Ajouter une colonne

```
ALTER TABLE nom_table ADD nom_colonne TYPE OPTIONS AFTER une_colonne;
ALTER TABLE matable ADD CA FLOAT NOT NULL AFTER age;
```

- Supprimer une colonne

```
ALTER TABLE nom_table DROP nom_colonne;
ALTER TABLE matable DROP CA;
```

- Modifier une colonne

```
ALTER TABLE nom_table CHANGE nom_colonne nouveau_nom_colonne TYPE OPTIONS;
ALTER TABLE matable CHANGE age age INT(3) NOT NULL;
```

STRUCTURE DES DONNÉES

DROP TABLE

```
DROP TABLE matable
```

STRUCTURE DES DONNÉES – LES INDEX

Permettent de stocker dans un arbre les différentes valeurs

- Les valeurs sont rangées triées
- Recherche dichotomique

Obligatoirement utilisés pour

- Les clés primaires
- Les clés étrangères

Plusieurs types d'index

INDEX	Autorise les doublons
UNIQUE	N'autorise pas les doublons
SPACIAL	Objets Géométriques
FULLTEXT	Objets de texte

STRUCTURE DES DONNÉES – LES INDEX

Créer un index

```
CREATE UNIQUE INDEX nom_index ON nom_table (colonne1, colonne2)
```

Supprimer un index

```
ALTER TABLE nom_table DROP INDEX nom_index
```

Voir les index d'une table

```
SHOW INDEX FROM nom_table;
```

STRUCTURE DES DONNÉES – LES CONTRAINTES

Les contraintes sont des index

Plusieurs types

PRIMARY KEY	Contrainte de clé primaire
FOREIGN KEY	Contrainte de clé étrangère
CHECK	Définir des règles de validation (valeurs booléennes)

STRUCTURE DES DONNÉES – LES CONTRAINTES

Créer une contrainte de clé primaire

```
ALTER TABLE nom_table
  ADD
    CONSTRAINT Nom_Index
    PRIMARY KEY (colonne1, colonne2)
```

```
ALTER TABLE matable
  ADD
    CONSTRAINT PK_MATABLE
    PRIMARY KEY (id)
```

STRUCTURE DES DONNÉES – LES CONTRAINTES

Créer une contrainte de clé étrangère

```
ALTER TABLE nom_table
  ADD
    CONSTRAINT Nom_Index
    FOREIGN KEY (colonne_table)
    REFERENCES nom_table_reference(colonne_table_reference)
```

```
ALTER TABLE matable
  ADD
    CONSTRAINT FK_ParentEnfants
    FOREIGN KEY (id_parent)
    REFERENCES
    matable_reference(id)
```

Supprimer une contrainte de clé étrangère

```
ALTER TABLE nom_table
DROP
FOREIGN KEY nom_contrainte
```

STRUCTURE DES DONNÉES – LES CONTRAINTES

Les contraintes de clés étrangères sont par défaut strict

- La suppression d'une donnée référencée n'est pas autorisée
- La modification d'un ID référencé n'est pas autorisé

- ON "ACTION"
  - CASCADE
  - RESTRICT
  - SET NULL

```
ALTER TABLE matable
  ADD
    CONSTRAINT FK_ParentEnfants
    FOREIGN KEY (id_parent)
    REFERENCES matable(id)
    ON DELETE CASCADE
    ON UPDATE RESTRICT;
```

SIRUCTUREDESDONNÉES

Créer les contraintes à la création de la table

```sql
CREATE TABLE nom_db.matable (
  id INT NOT NULL AUTO_INCREMENT,
  nom VARCHAR(100) NOT NULL,
  prenom VARCHAR(100) NOT NULL,
  age INT(3) NOT NULL,
  id_parent INT NOT NULL,
  PRIMARY KEY (id),
  INDEX (id_parent),
  UNIQUE (nom)
) ENGINE = InnoDB;
```

EXERCICE

Créer un index « Clé primaire » sur les deux tables

Créer une table « achat »

- ID_CLIENT, ID_PRODUIT, DATE, PRIX
- Clé primaire sur les 2 champs
- Clé étrangère sur les 2 champs

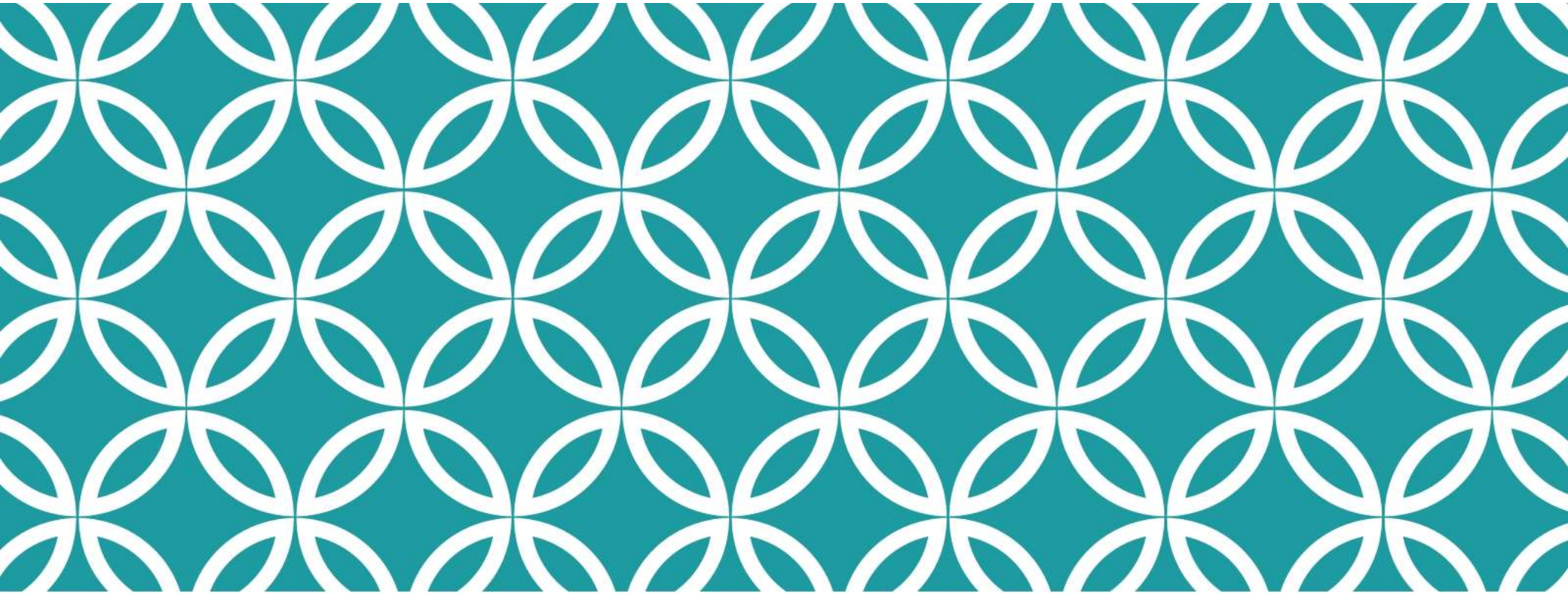
STRUCTURE DES DONNÉES – CHANGER MOTEUR

Changer le moteur de stockage (Storage Engine)

```
ALTER TABLE nom_table ENGINE = Moteur;

ALTER TABLE nom_table ENGINE = MyISAM;

ALTER TABLE nom_table ENGINE = InnoDB;
```



INTERROGATION DES DONNÉES

Langage DML

INTERROGATION DES DONNÉES

Langage DML (Data Manipulation Language)

Langage d'interrogation et de manipulation des données

CRUD

- Create
- Read
- Update
- Delete

INTERROGATION DES DONNÉES

INSERT INTO (C)

- Ajouter des données

SELECT (R)

- Sélectionner des données

UPDATE (U)

- Mettre à jour des données

DELETE (D)

- Supprimer des données

INTERROGATION DES DONNÉES

INSERT INTO

```
INSERT INTO ma_table (colonne1, colonne2) VALUES ('valeur 1', 'valeur 2');
```

INSERT INTO, valeurs multiples en MySql

```
INSERT INTO ma_table (colonne1, colonne2)
         VALUES ('valeur 1', 'valeur 2')
                , ('valeur 1', 'valeur 2')
                , ('valeur 1', 'valeur 2');
```

EXERCICE

Insérer quelques données dans les tables

- Adresse
- Personne
- Client
- Produit
- Commande

INTERROGATION DES DONNÉES

SELECT

```
SELECT colonne1, colonne2 FROM ma_table
```

```
SELECT * FROM ma_table
```

SELECT et ALIAS

```
SELECT colonne1 AS COL1, colonne2 AS COL2 FROM ma_table t
```

EXERCICE

Sélectionner toutes les personnes

Sélectionner le nom et le prix de tous les produits

Sélectionner toutes les commandes

INTERROGATION DES DONNÉES

SELECT et restriction WHERE

Signification	Opérateur
Egal à	=
Différent de	!= (ou <>)
Strictement supérieur à	>
Supérieur ou égal à	>=
Strictement inférieur à	<
Inférieur ou égal à	<=
Contient	LIKE '%val%'
Est	IS (ou <=>)
Dans une liste	IN

```
SELECT colonne1, colonne2
FROM ma_table
WHERE colonne1 = 'valeur'
```

```
SELECT colonne1, colonne2
FROM ma_table
WHERE
  (colonne1 = 'valeur' OR colonne1 = 'valeur 2')
  AND colonne2 = 'valeur 3'
```

Type de logique	Opérateur
ET	AND
OU	OR
NON	NOT

EXERCICE

Sélectionner la personne ID 1

Sélectionner le produit dont le nom est égal à une valeur

Sélectionner les produits contenant « a »

Sélectionner la personne ID 1 ET la personne ID 2

- Avec AND/OR et IN

INTERROGATION DES DONNÉES

Jointures

ID	LIBELLE	PRIX
1	GoPRO HERO 5	429.99
2	GoPRO KARMA	699.99

ID	NOM	PRENOM
1	PERROUAULT	Jérémy
2	PERROUAULT	Alissa
3	CESBRON	Martin

ID	LIBELLE	PRIX	ID	NOM	PRENOM
1	GoPRO HERO 5	429.99	1	PERROUAULT	Jérémy
2	GoPRO KARMA	699.99	1	PERROUAULT	Jérémy
1	GoPRO HERO 5	429.99	2	PERROUAULT	Alissa
2	GoPRO KARMA	699.99	2	PERROUAULT	Alissa
1	GoPRO HERO 5	429.99	3	CESBRON	Martin
2	GoPRO KARMA	699.99	3	CESBRON	Martin

EXERCICE

Sélectionner les achats de la personne 1

- Avec les informations de la personne

Sélectionner les produits ID 1 achetés

- Avec les informations du produit

INTERROGATION DES DONNÉES

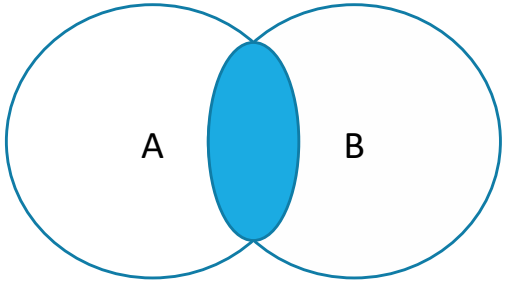
Jointures

- INNER JOIN (JOIN)
- LEFT JOIN (LEFT OUTER JOIN)
- RIGHT JOIN (RIGHT OUTER JOIN)
- FULL JOIN (FULL OUTER JOIN)

INTERROGATION DES DONNÉES

INNER JOIN (JOIN)

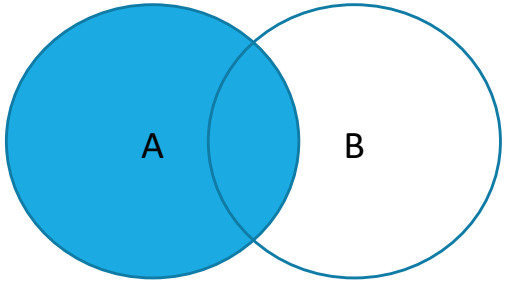
```sql
SELECT colonne1, colonne2
FROM table1 a
INNER JOIN table2 b ON b.col = a.col
```



INTERROGATION DES DONNÉES

LEFT JOIN (LEFT OUTER JOIN)

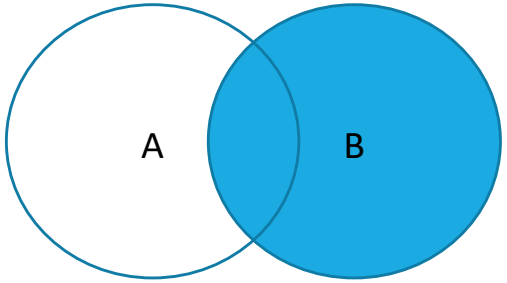
```sql
SELECT colonne1, colonne2
FROM table1 a
LEFT JOIN table2 b ON b.col = a.col
```



INTERROGATION DES DONNÉES

RIGHT JOIN (RIGHT OUTER JOIN)

```
SELECT colonne1, colonne2
FROM table1 a
RIGHT JOIN table2 b ON b.col = a.col
```

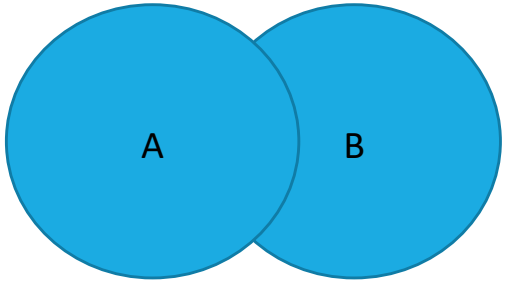


INTERROGATION DES DONNÉES

FULL JOIN (FULL OUTER JOIN)

Ne fonctionne pas sous MySQL

```
SELECT colonne1, colonne2
FROM table1 a
FULL JOIN table2 b ON b.col = a.col
```



INTERROGATION DES DONNÉES

Jointures

```sql
SELECT a.colonne1 As NomPersonnalise1, a.colonne2 As NomPersonnalise2
FROM table1 a
LEFT JOIN table2 b ON b.col = a.col
LEFT JOIN table3 c ON c.col = a.col
LEFT JOIN table4 d ON d.col = b.col

WHERE a.colonne2 = 'valeur'
```

INTERROGATION DES DONNÉES

DISTINCT

```
SELECT DISTINCT a.colonne1, a.colonne2
FROM table1 a
```

EXERCICE

Sélectionner les désignations

- Avec les informations de la personne
- Avec les informations du produit

Sélectionner toutes les personnes, y compris ceux qui n'ont rien commandé, et leurs achats le cas échéant

- Avec les informations du produit, si disponible

Sélectionner uniquement les personnes ayant effectuées des achats

- Sans les informations du produit

INTERROGATION DES DONNÉES

UPDATE

```
UPDATE ma_table
SET
  colonne1 = 'valeur 1',
  colonne2 = 'valeur 2'

WHERE colonne = 'valeur';
```

Sql Server / MySql

```
UPDATE ma_table
         inner Join autre_table On ma_table.champA =
autre_table.champB
SET
  colonne1 = 'valeur 1',
  colonne2 = 'valeur 2'

WHERE colonne = 'valeur';
         And autre_table.ChampC = 'valeur recherche'
```

MySql

EXERCICE

Modifier le nom de la personne ID 1

Modifier le prix du produit 1

- Augmentation du prix de 10 %

Modifier le prix des produits qui n’ont pas été achetés

- Diminution du prix de 20 %

INTERROGATION DES DONNÉES

DELETE

```sql
DELETE FROM ma_table
WHERE colonne = 'valeur';
```

EXERCICE

Supprimer le produit 1

Supprimer les produits qui n’ont pas été achetés

INTERROGATION DES DONNÉES – ORDER BY

ORDER BY

- Permet de ranger les informations par ordre croissant ou décroissant

```sql
SELECT colonne1, colonne2
FROM ma_table
ORDER BY
  colonne1 ASC,
  colonne2 DESC;
```

EXERCICE

Sélectionner tous les clients par ordre alphabétique (Prénom puis Nom)

INTERROGATION DES DONNÉES – LIMIT

LIMIT

- Permet de sélectionner une rangée d'informations

```
SELECT colonne1, colonne2
FROM ma_table
LIMIT start, maxi
```

- Les 30 premiers

```
SELECT colonne1, colonne2
FROM ma_table
LIMIT 0, 30
```

- Les 30 suivants

```
SELECT colonne1, colonne2
FROM ma_table
LIMIT 30, 30
```

EXERCICE

Sélectionner les 2 derniers achats

- Avec les informations client, produit et fournisseur

INTERROGATION DES DONNÉES – FONCTIONS

Fonctions d'agrégation

- AVG()
  ```
  SELECT AVG(colonne1) FROM ma_table;
  ```
  - Moyenne d'une colonne
- SUM()
  ```
  SELECT SUM(colonne1) FROM ma_table;
  ```
  - Somme d'une colonne
- MIN()
  - Minimum d'une colonne
- MAX()
  - Maximum d'une colonne
- COUNT()
  ```
  SELECT COUNT(colonne1) FROM ma_table;
  ```
  - Compter le nombre (selon une colonne)

INTERROGATION DES DONNÉES – FONCTIONS

Fonctions

- GROUP BY
  - Permet de regrouper par colonne
- HAVING
  - Remplace la clause WHERE dans le cas de restriction sur fonction d'agrégation

EXERCICE

Sélectionner tous les clients et leur CA

- Ranger les informations par CA décroissant

INTERROGATION DES DONNÉES

Possible de créer des requêtes imbriquées « sous-requêtes »

- Comme une table (FROM, JOIN, …)

```
SELECT tb.colonne3 FROM (SELECT colonne3, colonne4 FROM ma_table) tb
```

- Comme un champ

```
SELECT
  colonne1,
  (
    SELECT MIN(colonne2)
    FROM ma_table2
  ) AS col
FROM ma_table t
```

- Dans une clause WHERE

```
SELECT colonne1
FROM ma_table t
WHERE colonne2 IN (SELECT colonne3 FROM ma_table2)
```

EXERCICE

Sélectionner tous les clients et leur CA

- Uniquement ceux dont le CA est compris entre 100 et 200 euros

Compter le nombre de clients dont le CA est supérieur à 200 euros

Sélectionner tous les clients et le nombre de produits uniques achetés

EXERCICE

Sélectionner les clients

- Le prix minimum d'un produit acheté, et son libellé
- Le prix maximum d'un produit acheté, et son libellé
- Son panier moyen
- Sa première date d'achat
- Sa dernière date d'achat

Fonctions d'agrégation

- AVG()
  - Moyenne d'une colonne
- SUM()
  - Somme d'une colonne
- MIN()
  - Minimum d'une colonne
- MAX()
  - Maximum d'une colonne
- COUNT()
  - Compter le nombre (selon une colonne)

Fonctions

- GROUP BY
  - Permet de regrouper par colonne
- HAVING
  - Remplace la clause WHERE dans le cas de restriction sur fonction d'agrégation