# Base de Données

Session 1
Généralité
SQL débutant

Un cours sur le langage SQL n’est vraiment utile que si on essaye de le mettre en pratique dans un contexte d’usage réel.

# Les différents type de Base de données

- Les Bases De Données dites “Relationnelles”

- Les Bases De Données dites “NoSQL”

# Les Bases De Données dites “Relationnelles”

- Ex : MySQL, PostgreSQL, Oracle, Microsoft SQL Server ...

- Permet de mettre en avant les relations entre les données.
  **Ces données sont organisées en table dans des lignes et colonnes** afin d’être accessibles.

- La relation entre les **tables** peut ensuite être définie à l'aide de **clés étrangères.**
  Une **clé étrangère** est un champ d'une table qui est lié à la **clé primaire** d'une autre table.

# Contraintes du modèle relationnel

- **Les contraintes d'intégrité relationnelle se réfèrent aux conditions qui doivent être présentes pour qu'une relation soit valide**

    - **Contraintes de clé** indiquent qu'une table doit toujours avoir une clé primaire
    - **Contraintes de domaine** limitent la plage des valeurs de domaine d'un attribut
    - **Contraintes d'intégrité référentielle** indiquent que les relations entre les tables doivent toujours être cohérentes

# BDD relationnelle :: Avantages et Inconvénients

- **Avantages :**
  - **Indépendance structurelle**
    => La base de données relationnelle ne concerne que les données et non une structure.
  - **Facilité d'utilisation**
    => Le modèle relationnel est très intuitif à utiliser car composé de tableaux organisés de lignes et de colonnes.
  - **Capacité d'interrogation**
    => Il permet à un langage de requête de haut niveau comme SQL d'éviter une navigation complexe dans la base de données.
  - **Indépendance des données**
    => La structure d'une base de données peut être modifiée sans avoir à changer d'application.
  - **Redondance des données**
    => Une base de données relationnelle garantit qu'aucun attribut n'est répété.

# BDD relationnelle :: Avantages et Inconvénients

- **Inconvéniants :**
  - **Taille**
    => Certaines bases de données relationnelles ont des limites sur la longueur des champs utilisés.

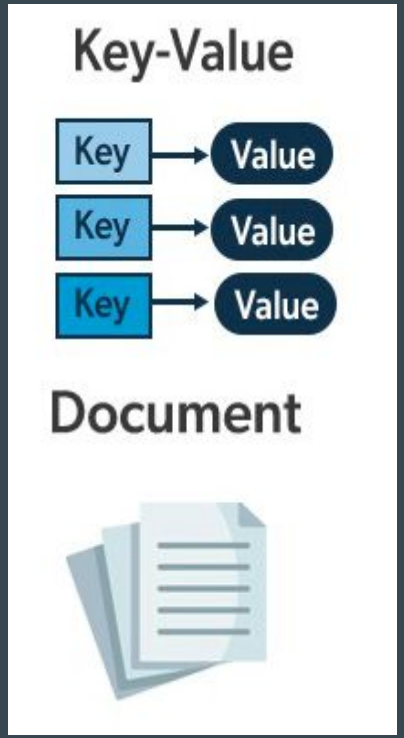
  - **Complexité**
    => Peuvent parfois devenir complexes à mesure que la quantité de données augmente et que les relations entre les éléments de données deviennent plus complexes.

  - **Scalabilité**
    => Elles sont quasiment toute limités à une scalabilité Verticale

# Les différents type de Base de données NoSQL

- Base de type “Clés - Valeurs”
  - Cache : Redis, MemCached, Couchbase, Hazelcast...
  - Stocké : ArangoDB, Couchbase

- Base de type “Documents” :
  - MongoDB, ElasticSearch, Couchbase, CouchDB, ArangoDB

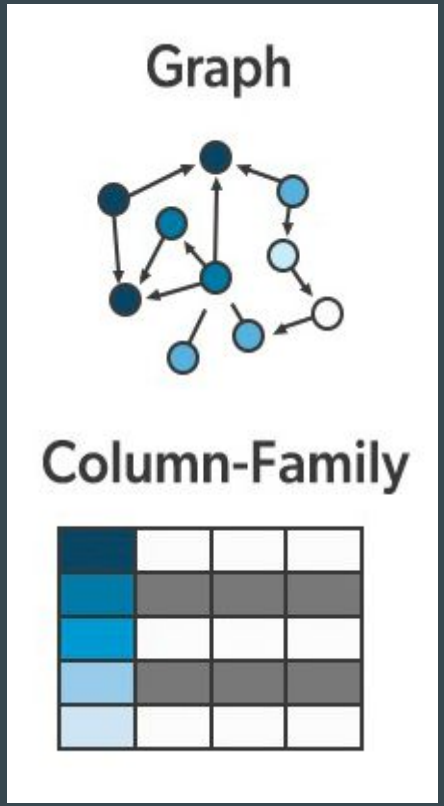



https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML

# Les différents type de Base de données NoSQL

- Base orienté de type “Graph” (Neo4J)
  - Neo4J, ArangoDB ...

- Base orienté de type “Colonne”
  - Cassandra, HBase, Bigtable



# Performance

| Data model | Performance | Scalability | Flexibility | Complexity | Functionality |
|---|---|---|---|---|---|
| Key–value store | high | high | high | none | variable (none) |
| Column-oriented store | high | high | moderate | low | minimal |
| Document-oriented store | high | variable (high) | high | low | variable (low) |
| Graph database | variable | variable | high | high | graph theory |
| Relational database | variable | variable | low | moderate | relational algebra |

# SQL Structured Query Language

- SELECT
- UPDATE
- DELETE
- INSERT

# SELECT

```
SELECT nom_du_champ FROM nom_du_tableau;
```

```
SELECT * FROM client;
```

| identifiant | prenom | nom | ville |
|---|---|---|---|
| 1 | Pierre | Dupond | Paris |
| 2 | Sabrina | Durand | Nantes |
| 3 | Julien | Martin | Lyon |
| 4 | David | Bernard | Marseille |
| 5 | Marie | Leroy | Grenoble |

# SELECT

```
SELECT ville FROM client;
```

| ville |
| --- |
| Paris |
| Nantes |
| Lyon |
| Marseille |
| Grenoble |

```
SELECT prenom, nom FROM client;
```

| prenom | nom |
| --- | --- |
| Pierre | Dupond |
| Sabrina | Durand |
| Julien | Martin |
| David | Bernard |
| Marie | Leroy |

# WHERE

```sql
SELECT nom_colonnes FROM nom_table WHERE conditions;
```

# WHERE

| id | nom | nbr_commande | ville |
|---|---|---|---|
| 1 | Paul | 3 | paris |
| 2 | Maurice | 0 | rennes |
| 3 | Joséphine | 1 | toulouse |
| 4 | Gérard | 7 | paris |

```
SELECT * FROM client WHERE ville = 'paris';
```

| id | nom | nbr_commande | ville |
|---|---|---|---|
| 1 | Paul | 3 | paris |
| 4 | Gérard | 7 | paris |

# SQL :: Opérateurs de comparaisons

| Opérateur | Description |
|---|---|
| = | Égal |
| <> | Pas égal |
| != | Pas égal |
| > | Supérieur à |
| < | Inférieur à |
| >= | Supérieur ou égal à |
| <= | Inférieur ou égal à |
| IN | Liste de plusieurs valeurs possibles |
| BETWEEN | Valeur comprise dans un intervalle donnée (utile pour les nombres ou dates) |
| LIKE | Recherche en spécifiant le début, milieu ou fin d'un mot. |
| IS NULL | Valeur est nulle |
| IS NOT NULL | Valeur n'est pas nulle |

# AND/OR

```sql
SELECT nom_colonnes
FROM nom_table
WHERE condition1 AND condition2;
```

```sql
SELECT nom_colonnes
FROM nom_table
WHERE condition1 OR condition2;
```

# AND/OR

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 2 | clavier | informatique | 32 | 35 |
| 3 | souris | informatique | 16 | 30 |
| 4 | crayon | fourniture | 147 | 2 |

```
SELECT * FROM produit WHERE categorie = 'informatique' AND stock < 20;
```

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 3 | souris | informatique | 16 | 30 |

# AND/OR

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 2 | clavier | informatique | 32 | 35 |
| 3 | souris | informatique | 16 | 30 |
| 4 | crayon | fourniture | 147 | 2 |

```
SELECT * FROM produit WHERE nom = 'ordinateur' OR nom = 'clavier';
```

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 2 | clavier | informatique | 32 | 35 |

# AND/OR

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 2 | clavier | informatique | 32 | 35 |
| 3 | souris | informatique | 16 | 30 |
| 4 | crayon | fourniture | 147 | 2 |

```
SELECT *
FROM produit
WHERE (categorie = 'informatique' AND stock < 20)
    OR (categorie = 'fourniture'    AND stock < 200);
```

| id | nom | categorie | stock | prix |
|---|---|---|---|---|
| 1 | ordinateur | informatique | 5 | 950 |
| 2 | souris | informatique | 16 | 30 |
| 4 | crayon | fourniture | 147 | 2 |

# IN

```sql
SELECT nom_colonne
FROM table
WHERE nom_colonne IN ( valeur1, valeur2, valeur3, ... );
```

# IN

```sql
SELECT prenom
FROM utilisateur
WHERE prenom = 'Maurice' OR prenom = 'Bob' OR prenom = 'Lena';
```

```sql
SELECT prenom
FROM utilisateur
WHERE prenom IN ('Maurice', 'Bob', 'Lena');
```

# LIKE

| id | nom | ville |
|---|---|---|
| 1 | Léon | Lyon |
| 2 | Odette | Nice |
| 3 | Vivien | Nantes |
| 4 | Etienne | Lille |

```
SELECT * FROM client WHERE ville LIKE 'N%';
```

| id | nom | ville |
|---|---|---|
| 2 | Odette | Nice |
| 3 | Vivien | Nantes |

# LIKE

| id | nom | ville |
|---|---|---|
| 1 | Léon | Lyon |
| 2 | Odette | Nice |
| 3 | Vivien | Nantes |
| 4 | Etienne | Lille |

```
SELECT * FROM client WHERE ville LIKE '%e';
```

| id | nom | ville |
|---|---|---|
| 2 | Odette | Nice |
| 4 | Etienne | Lille |

# DISTINCT

```
SELECT DISTINCT nom_colonne FROM table;
```

| identifiant | prenom | nom |
|---|---|---|
| 1 | Pierre | Dupond |
| 2 | Sabrina | Bernard |
| 3 | David | Durand |
| 4 | Pierre | Leroy |
| 5 | Marie | Leroy |

```
SELECT DISTINCT prenom FROM client;
```

| prenom |
|---|
| Pierre |
| Sabrina |
| David |
| Marie |

# ORDER BY

```
SELECT colonne1, colonne2, colonne3
FROM table
ORDER BY colonne1 DESC, colonne2 ASC;
```

# ORDER BY

| id | nom | prenom | date_inscription | tarif_total |
|---|---|---|---|---|
| 1 | Durand | Maurice | 2012-02-05 | 145 |
| 2 | Dupond | Fabrice | 2012-02-07 | 65 |
| 3 | Durand | Fabienne | 2012-02-13 | 90 |
| 4 | Dubois | Chloé | 2012-02-16 | 98 |
| 5 | Dubois | Simon | 2012-02-23 | 27 |

```
SELECT * FROM utilisateur ORDER BY nom;
```

| id | nom | prenom | date_inscription | tarif_total |
|---|---|---|---|---|
| 4 | Dubois | Chloé | 2012-02-16 | 98 |
| 5 | Dubois | Simon | 2012-02-23 | 27 |
| 2 | Dupond | Fabrice | 2012-02-07 | 65 |
| 1 | Durand | Maurice | 2012-02-05 | 145 |
| 3 | Durand | Fabienne | 2012-02-13 | 90 |

# LIMIT / OFFSET

```
SELECT * FROM table LIMIT 10, OFFSET 5;
```

```
SELECT * FROM table LIMIT 5;
```

```
SELECT * FROM table LIMIT 5, 10;
```

# UPDATE

```
UPDATE table
SET colonne1 = 'valeur1', colonne2 = 'valeur2'
WHERE conditions;
```

# UPDATE

| id | nom | rue | ville | code_postal | pays |
|---|---|---|---|---|---|
| 1 | Chantal | 12 Avenue du Petit Trianon | Puteaux | 92800 | France |
| 2 | Pierre | 18 Rue de l'Allier | Ponthion | 51300 | France |
| 3 | Romain | 3 Chemin du Chiron | Trévérien | 35190 | France |

```
UPDATE client
SET rue = '49 Rue Ameline', ville = 'Saint-Eustache-la-Forêt', code_postal = '76210'
WHERE id = 2;
```

| id | nom | rue | ville | code_postal | pays |
|---|---|---|---|---|---|
| 1 | Chantal | 12 Avenue du Petit Trianon | Puteaux | 92800 | France |
| 2 | Pierre | 49 Rue Ameline | Saint-Eustache-la-Forêt | 76210 | France |
| 3 | Romain | 3 Chemin du Chiron | Trévérien | 35190 | France |

# DELETE

```
DELETE FROM `table` WHERE conditions;
```

# DELETE

| id | nom | prenom | date_inscription |
|---|---|---|---|
| 1 | Bazin | Daniel | 2012-02-13 |
| 2 | Favre | Constantin | 2012-04-03 |
| 3 | Clerc | Guillaume | 2012-04-12 |

```
DELETE FROM `utilisateur` WHERE id = 1;
```

| id | nom | prenom | date_inscription |
|---|---|---|---|
| 2 | Favre | Constantin | 2012-04-03 |
| 3 | Clerc | Guillaume | 2012-04-12 |

# INSERT INTO

```
INSERT INTO `table` VALUES ('valeur 1', 'valeur2', '...' );
```

```
INSERT INTO `table` (nom_colonne1, nom_colonne2, ... )
VALUES ('valeur 1', 'valeur2', '...' );
```

# INSERT INTO

```
INSERT INTO `client` (prenom, nom, ville, age)
VALUES
 ('Rébecca', 'Armand', 'Saint-Didier-des-Bois', 24),
 ('Aimée', 'Hebert', 'Marigny-le-Châtel', 36),
 ('Marielle', 'Ribeiro', 'Maillères', 27),
 ('Hilaire', 'Savary', 'Conie-Molitard', 58);
```

| id | prenom | nom | ville | age |
|---|---|---|---|---|
| 1 | Rébecca | Armand | Saint-Didier-des-Bois | 24 |
| 2 | Aimée | Hebert | Marigny-le-Châtel | 36 |
| 3 | Marielle | Ribeiro | Maillères | 27 |
| 4 | Hilaire | Savary | Conie-Molitard | 58 |

# Le Castor de l'enfer et DM

1. Obtenir la liste des 10 villes les plus peuplées en 2012
2. Obtenir la liste des 50 villes ayant la plus faible superficie
3. Obtenir la liste des départements d'outres-mer, c'est-à-dire ceux dont le numéro de département commencent par "97"
4. (bonus) Compter le nombre de villes dont le nom commence par "saint"

```
SELECT COUNT(*) FROM table;
```

5. (bonus) Remplacez les tirets par un espace vide, pour toutes les villes commençant par "SAINT-" (dans la colonne qui contient les noms en majuscule)

```
SELECT REPLACE('Hello le monde', 'Hello', 'Bonjour') FROM utilisateur 1;
```