SQL & SGBD

Introduction aux SQLs

1 – BDD
Définition / Explication

2 – BDDR vs NoSQL
Différences et choix

3 – Les différents SQL
Une multitude de choix

4 – Modélisation
MCD, MLD, MPD

5 – Concept ACID/BASE
Pas de chimie ici

6 – Optimisation et Sécurité
Performance, scalabilité et défense

01

Base de données ? SGBD ?

Base de Données ? Comment et Pourquoi

BDD != SGBD

Base De Données != Système de Gestion de Base de Données

- BDD = l'ensemble des données
- SGBD = L'outil permettant de gérer les données

BDD en DevWeb = Crucial !

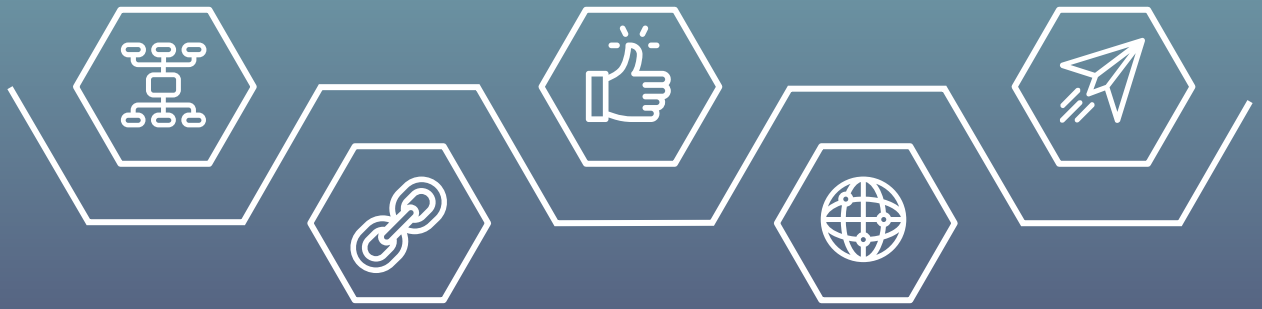
Structuration des données

Sécurité et sauvegarde

Facilité d'accès via des requêtes

Mutualisation et accès simultané

Accès rapide aux données



02

BDD Relationnelle vs NoSQL

Bases de Données Relationnelles

- **Base**
  - Tables
    - Colonnes (champs)
      - Lignes

- **Schéma rigide**
- **Plusieurs SGBD**
  - MySQL
  - PostgreSQL
  - SQLite
  - Oracle
  - etc.

Bases de Données Relationnelles

- Structure claire et définie
- Intégrité des données assurée
- Possibilités d'opérations complexes
- Transactions ACID (Atomicité, Cohérence, Isolation, Durabilité)

- Difficulté à gérer des données non structurées
- Moins adaptées à l'évolutivité horizontale
- Manque d'efficacité à grande échelle

Bases de Données NoSQL

- **Schéma souple**
- **Différents types de NoSQL**
  - Document (MongoDB, CouchDB)
  - Graph (Neo4j, Amazon Neptune)
  - Column (Cassandra, HBase)
  - Key-Value (Redis, DynamoDB)

Bases de Données NoSQL

- Flexibilité de schéma
- Scalabilité horizontale
- Gestion de données massives - Big Data
- Transaction BASE (Basically Available, Soft state, Eventual consistency)

- Moins adaptées aux opérations complexes
- Nécessite des connaissances spécifiques pour chaque NoSQL
- Pas très développées en terme de fonctionnalités

Comparaison BDDR et NoSQL

Caractérique	BDDR (Relationnelle)	NoSQL (Non-relationnelle
STRUCTURE	Tables avec schéma fixe	Schéma flexible
LANGAGE DE REQUÊTE	SQL	Propre à chaque système
TRANSACTIONS	ACID	BASE
SCALABILITÉ	Verticale	Horizontale
USAGE TYPIQUE	Transactions complexes, système de gestion	Big Data

03

Les différents types de SQL

Deux catégories de SQL

Client lourd

- Infrastructure robuste
- Fonctionnalités avancées
- Coût élevé
- Maintenance et gestion

Oracle
SQL Server

Client léger

- Infrastructure légère
- Simplicité de déploiement
- Coût faible
- Bonne performance

MySQL
SQLite

04

Modélisation d'une BDD

Processus de modélisation

MCD

Modèle **C**onceptuel de **D**onnées

MLD

Modèle **L**ogique de **D**onnées

MPD

Modèle **P**hysique de **D**onnées

Le Modèle Conceptuel de Données

ENTITÉS

Les acteurs ou objets de la BDD avec leurs attributs

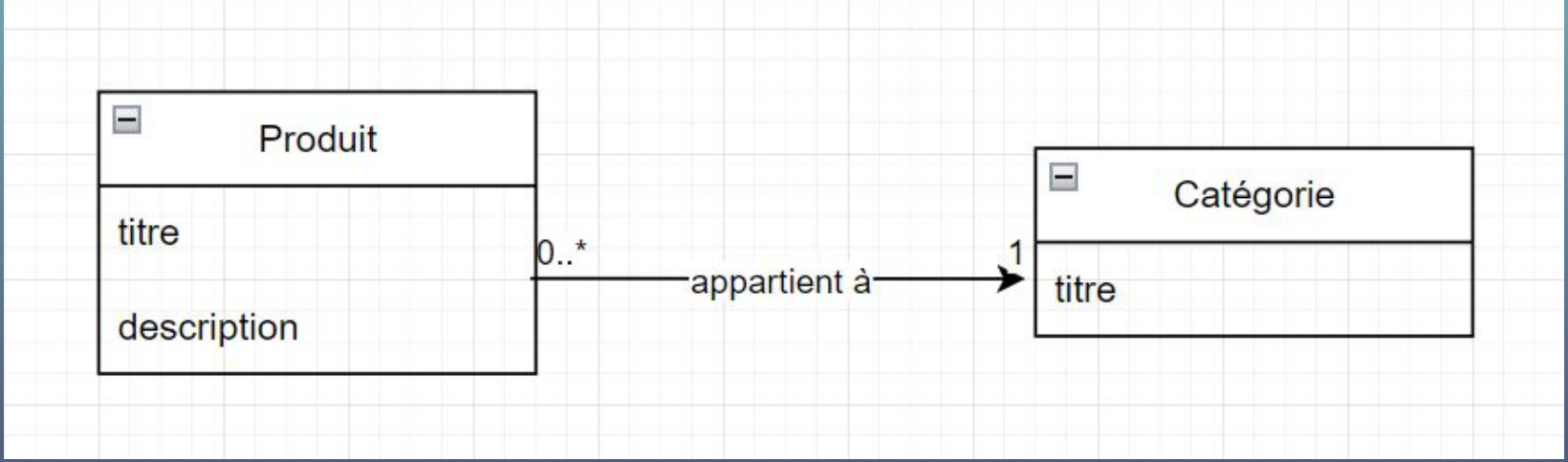
ASSOCIATIONS

Les relations entre ces entités

CARDINALITÉS

Indique la nature de la relation

Le Modèle Conceptuel de Données



Le Modèle Logique de Données

TABLES

Les entités deviennent des tables, les attributs des colonnes

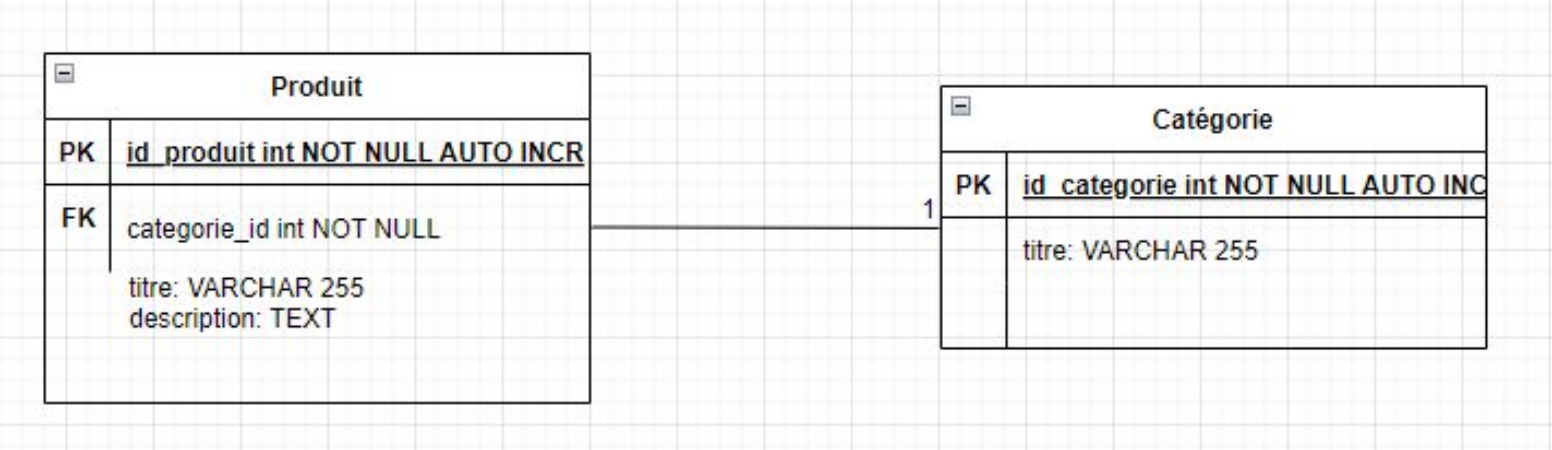
CLÉS PRIMAIRES

Chaque table doit avoir une clé primaire (id unique)

CLÉS ÉTRANGÈRES

Ce sont les références vers les autres tables

Le Modèle Logique de Données



Le Modèle Physique de Données

TRANSCRIPTION

Les schémas deviennent du code de création des tables

TYPES DES DONNÉES

Association de chaque colonne au type correspondant dans le SQL

CONTRAINTES

Spécificités des champs (NOT NULL, UNIQUE, AI, etc.)

Le Modèle Physique de Données

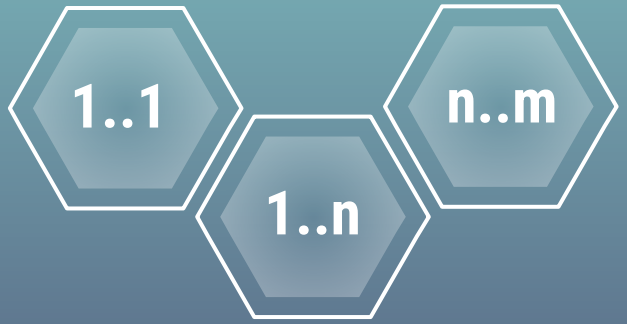
```sql
CREATE TABLE Categorie (
    id_categorie INT AUTO_INCREMENT PRIMARY KEY,
    titre VARCHAR(255) NOT NULL
);

CREATE TABLE Produit (
    id_produit INT AUTO_INCREMENT PRIMARY KEY,
    titre VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    categorie_id INT,
    FOREIGN KEY (categorie_id) REFERENCES Categorie(id_categorie)
);
```

Les cardinalités

OneToOne

Relation de 1 à 1



ManyToMany

Relation de plusieurs à plusieurs

ManyToOne – OneToMany

Relation de 1 à plusieurs

05

Concepts ACID vs BASE

Concept ACID = BDD Relationnelle

Atomicité

Une transaction est indivisible

Cohérence

Une transaction validée = état cohérent

Isolation

Les transactions sont indépendantes

Durabilité

Une transaction validée est permanente

Concept BASE = BDD NoSQL

Basically Available

Données toujours dispo mais cohérence pas immédiate

Soft state

L'état du système peut changer même sans requêtes

Eventual Consistency

La cohérence sera toujours retrouvée

06

Optimisation et sécurité

Optimisation

- **Indexation**
  - Accélère les requêtes de lecture basées sur un champ mais ralenti les écritures
- **Clés étrangères**
  - Évitent les incohérences dans les données en garantissant l'intégrité référentielle
- **Normalisation**
  - Évite la redondance de l'information mais peut complexifier certaines requêtes

Sécurité

- **Permissions**
  - Limitation des accès en fonction des rôles des utilisateurs
- **Injection SQL**
  - Utilisation de requêtes préparées pour éviter les attaques
- **Chiffrement**
  - Crypter les données sensibles comme les mots de passe
- **Logs et surveillance**
  - Suivre les accès et les modifications suspectes
- **Pare-feu applicatif**
  - Blocage des intrusions

En avant !

Mathieu Quittard mathieu.qtrd@gmail.com