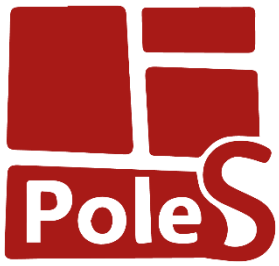


Le SQL

Sommaire

1. Qu'est-ce que SQL ?

SQL (Structured Query Language) est un langage de programmation utilisé pour gérer et manipuler des bases de données relationnelles. Il permet de créer, modifier, interroger et gérer des données dans un Système de Gestion de Base de Données (SGBD).

Histoire :

En juin 1970, Edgar Frank Codd publia l'article A Relational Model of Data for Large Shared Data Banks (« Un référentiel de données relationnel pour de grandes banques de données partagées ») dans la revue Communications of the ACM (Association for Computing Machinery). Ce référentiel relationnel fondé sur la logique des prédicats du premier ordre a été rapidement reconnu comme un modèle théorique intéressant, pour l'interrogation des bases de données, et a inspiré le développement du langage Structured English QUEry Language (SEQUEL) (« langage d'interrogation structuré en anglais »), renommé ultérieurement SQL pour cause de conflit de marque déposée.

Développée chez IBM en 1970 par Donald Chamberlin et Raymond Boyce, cette première version a été conçue pour manipuler et éditer des données stockées dans la base de données relationnelle à l'aide du système de gestion de base de données IBM System R. Le nom SEQUEL, qui était déposé commercialement par l'avionneur Hawker Siddeley pour un système d'acquisition de données, a été abandonné et contracté en SQL en 19751. SQL était censé alors devenir un élément clé du futur projet FS.

En 1979, Relational Software, Inc. (actuellement Oracle Corporation) présenta la première version commercialement disponible de SQL rapidement imité par d'autres fournisseurs.

SQL a été adopté comme recommandation par l'Institut de normalisation américaine (ANSI) en 1986, puis comme norme internationale par l'ISO en 1987 sous le nom de ISO/CEI 9075 - Technologies de l'information - Langages de base de données - SQL2.

2. Qu'est-ce que SGBD ?

Un Système de Gestion de Base de Données (SGBD) est un logiciel qui permet de stocker des informations dans une base de données. Un tel système permet de lire, écrire, modifier, trier, transformer ou même imprimer les données qui sont contenus dans la base de données.

Parmi les logiciels les plus connus il est possible de citer : MySQL, PostgreSQL, SQLite, Oracle Database, Microsoft SQL Server, Firebird ou Ingres.

Rank			DBMS	Database Model	Score		
Jun 2023	May 2023	Jun 2022			Jun 2023	May 2023	Jun 2022
1.	1.	1.	Oracle	Relational, Multi-model	1231.48	-1.16	-56.27
2.	2.	2.	MySQL	Relational, Multi-model	1163.94	-8.52	-25.27
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	930.06	+9.97	-3.76
4.	4.	4.	PostgreSQL	Relational, Multi-model	612.82	-5.08	-8.02
5.	5.	5.	MongoDB	Document, Multi-model	425.36	-11.25	-55.36
6.	6.	6.	Redis	Key-value, Multi-model	167.35	-0.78	-7.96
7.	7.	7.	IBM Db2	Relational, Multi-model	144.89	+1.87	-14.30
8.	8.	8.	Elasticsearch	Search engine, Multi-model	143.75	+2.11	-12.25
9.	↑ 10.	9.	Microsoft Access	Relational	134.45	+3.28	-7.36
10.	↓ 9.	10.	SQLite	Relational	131.21	-2.65	-4.22
11.	11.	↑ 13.	Snowflake	Relational	114.13	+2.41	+17.71
12.	12.	↓ 11.	Cassandra	Wide column	108.55	-2.58	-6.90
13.	13.	↓ 12.	MariaDB	Relational, Multi-model	97.31	+0.44	-14.27
14.	14.	14.	Splunk	Search engine	89.45	+2.81	-6.11
15.	15.	↑ 16.	Amazon DynamoDB	Multi-model	79.90	-1.20	-3.98
16.	16.	↓ 15.	Microsoft Azure SQL Database	Relational, Multi-model	78.96	-0.23	-7.05
17.	17.	17.	Hive	Relational	75.52	+1.91	-6.06
18.	18.	↑ 24.	Databricks	Multi-model	65.82	+1.87	+17.69
19.	19.	↓ 18.	Teradata	Relational, Multi-model	62.64	-0.07	-7.76
20.	20.	↑ 23.	Google BigQuery	Relational	54.64	-0.24	+5.57

3. Qu'est-ce que MySQL ?

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, PostgreSQL et Microsoft SQL Server.

Son nom vient du prénom de la fille du cocréateur Michael Widenius, My (sv) (prononcer [my]). SQL fait référence au Structured Query Language, le langage de requête utilisé.

MySQL AB a été acheté le 16 janvier 2008 par Sun Microsystems pour un milliard de dollars américains4. En 2009, Sun Microsystems a été acquis par Oracle Corporation, mettant entre les mains d'une même société les deux produits concurrents que sont Oracle Database et MySQL. Ce rachat a été autorisé par la Commission européenne le 21 janvier 20105,6.

Depuis mai 2009, son créateur Michael Widenius a créé MariaDB (Maria est le prénom de sa deuxième fille) pour continuer son développement en tant que projet Open Source.

4. Qu'est-ce que phpMyAdmin ?

Il s'agit de l'une des plus célèbres interfaces pour gérer une base de données MySQL sur un serveur PHP. De nombreux hébergeurs, gratuits comme payants, le proposent ce qui évite à l'utilisateur d'avoir à l'installer.

Cette interface pratique permet d'exécuter, très facilement et sans grandes connaissances en bases de données, des requêtes comme les créations de table de données, insertions, mises à jour, suppressions et modifications de structure de la base de données, ainsi que l'attribution et la révocation de droits et l'import/export. Ce système permet de sauvegarder commodément une base de données sous forme de fichier .sql et d'y transférer ses données, même sans connaître SQL.

5. Création d'une Base de Données et de Tables

Dans MySQL, la création d'une base de données se fait avec la commande CREATE DATABASE. La création de tables s'effectue avec CREATE TABLE, où vous définissez les colonnes et leurs types de données.

-- Création d'une base de données :

```
CREATE DATABASE ma_base_de_donnees;
```

-- Sélection de la base de données :

```
USE ma_base_de_donnees;
```

-- Création d'une table :

```
CREATE TABLE utilisateurs (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(50),
    age INT
);
```

6. Insertion, Mise à Jour et Suppression de Données

Les opérations CRUD (Create, Read, Update, Delete) sont utilisées pour manipuler les données. INSERT ajoute des données, UPDATE modifie des données existantes, et DELETE supprime des données.

-- Insertion de données :

```
INSERT INTO utilisateurs (nom, age) VALUES ('Alice', 25), ('Bob', 30);
```

-- Mise à jour de données :

```
UPDATE utilisateurs SET age = 26 WHERE nom = 'Alice';
```

-- Suppression de données :

```
DELETE FROM utilisateurs WHERE nom = 'Bob';
```

7. Sélection de Données avec SELECT

La commande SELECT est utilisée pour récupérer des données d'une table. Vous pouvez spécifier des critères de filtrage, trier les résultats, etc.

-- Sélection de toutes les colonnes de la table utilisateurs :

```
SELECT * FROM utilisateurs;
```

-- Sélection de données avec un critère :

```
SELECT * FROM utilisateurs WHERE age > 25;
```

-- Tri des résultats :

```
SELECT * FROM utilisateurs ORDER BY age DESC;
```

8. Jointures avec MySQL

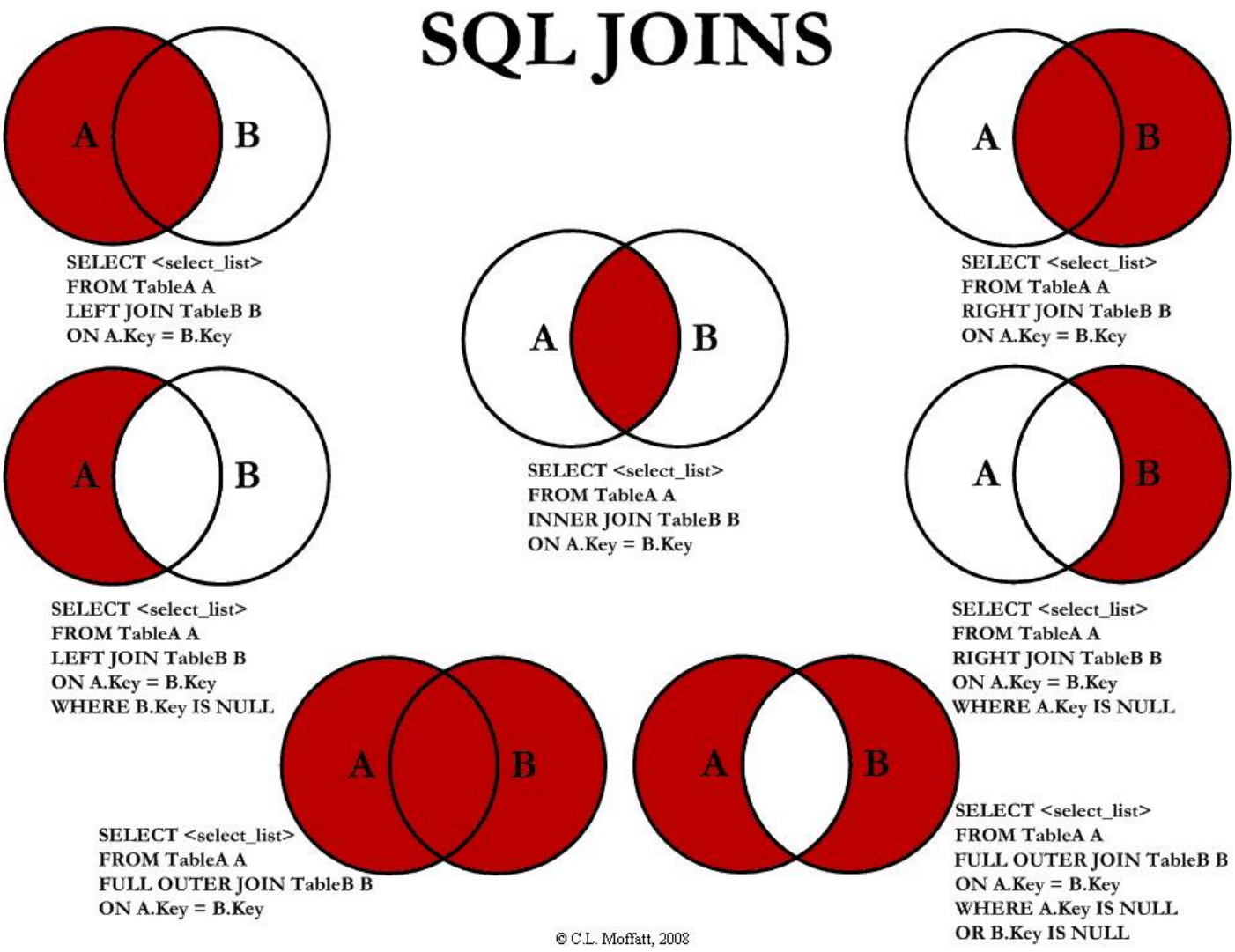
Les jointures (INNER JOIN, LEFT JOIN, RIGHT JOIN, etc.) sont utilisées pour combiner des lignes de deux ou plusieurs tables basées sur une condition de relation entre elles.

-- Création d'une deuxième table :

```
CREATE TABLE commandes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    utilisateur_id INT,
    produit VARCHAR(50),
    FOREIGN KEY (utilisateur_id) REFERENCES utilisateurs(id)
);
```

-- Requête INNER JOIN :

```
SELECT utilisateurs.nom, commandes.produit
FROM utilisateurs
INNER JOIN commandes ON utilisateurs.id = commandes.utilisateur_id;
```



9. Fonctions Agrégées avec MySQL

Les fonctions agrégées telles que SUM, AVG, COUNT, etc., sont utilisées pour effectuer des calculs sur un ensemble de valeurs.

-- Utilisation de fonctions agrégées :

```
SELECT AVG(age) AS age_moyen FROM utilisateurs;

SELECT COUNT(*) AS nombre_utilisateurs FROM utilisateurs;
```

10. Sous-requêtes avec MySQL

Les sous-requêtes sont des requêtes SQL imbriquées à l'intérieur d'une requête principale, utilisées pour récupérer des données plus complexes.

-- Utilisation d'une sous-requête :

```
SELECT nom
FROM utilisateurs
WHERE id IN (SELECT utilisateur_id FROM commandes WHERE produit = 'Livre');
```

11. Sécurité avec MySQL

La sécurité dans MySQL est essentielle pour protéger vos données contre les accès non autorisés et les vulnérabilités. Voici quelques bonnes pratiques pour sécuriser une base de données MySQL :

- Mots de passe forts :
  - *Utilisez des mots de passe forts pour les comptes MySQL.*
  - *Encouragez l'utilisation de caractères spéciaux, de chiffres et de lettres majuscules et minuscules.*
  - *Évitez d'utiliser des mots de passe évidents.*

- Privilèges appropriés :
  - *Accordez les privilèges les plus bas possibles aux utilisateurs (principe du moindre privilège).*
  - *Évitez d'utiliser des comptes avec des privilèges "SUPER" sauf si nécessaire.*

- Gestion des utilisateurs :
  - *Supprimez les comptes d'utilisateurs inutiles.*
  - *Évitez d'utiliser le compte root pour des opérations quotidiennes. Créez des utilisateurs spécifiques avec des privilèges appropriés.*

- Mises à jour régulières :
  - *Assurez-vous de maintenir à jour votre serveur MySQL avec les derniers correctifs de sécurité.*

- Audit et suivi :
    - *Activez les fonctionnalités d'audit pour suivre les activités sur la base de données.*
    - *Surveillez les journaux d'erreurs MySQL pour détecter toute activité suspecte.*

- Gestion des sauvegardes :
    - *Effectuez régulièrement des sauvegardes de votre base de données.*
    - *Testez la restauration des sauvegardes pour vous assurer qu'elles sont fonctionnelles.*

- Validation des entrées :
    - *Utilisez des requêtes paramétrées pour prévenir les attaques par injection SQL.*
    - *Validez et nettoyez toutes les données utilisateur avant de les utiliser dans des requêtes SQL.*

- Gestion des erreurs :
    - *Limitez l'information renvoyée par les erreurs MySQL pour ne pas exposer des détails sensibles.*

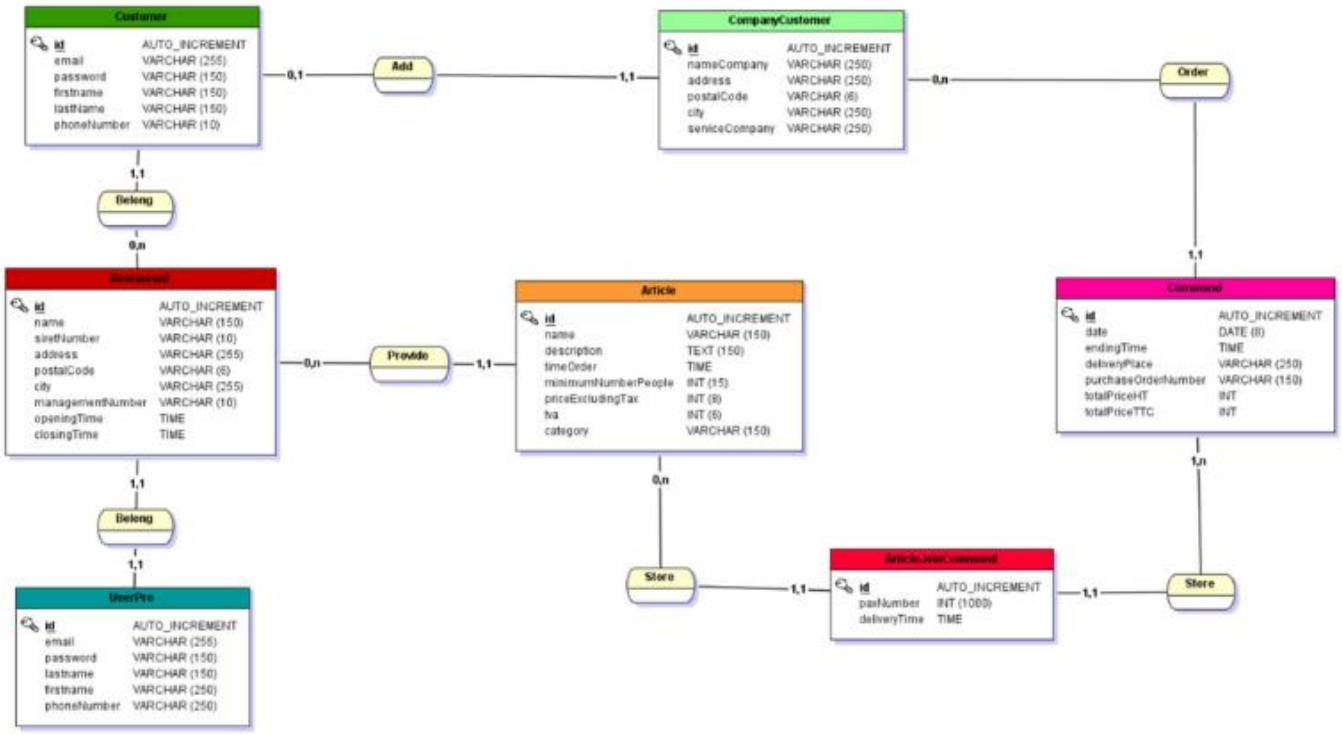
12. Introduction aux Diagrammes MCD et MLD

Les diagrammes MCD et MLD sont des outils de modélisation utilisés pour concevoir et représenter la structure d'une base de données.

13. Création d'un Modèle Conceptuel de Données (MCD)

Le MCD représente les concepts et les relations métier d'un système sans se soucier des détails de la mise en œuvre technique. C'est une représentation visuelle des entités (objets ou concepts) et de leurs relations.

Il a été créé pour faciliter la compréhension et la communication entre les parties prenantes non techniques d'un projet (comme les clients ou les responsables métier) et les développeurs. Il permet de modéliser les informations de manière abstraite.



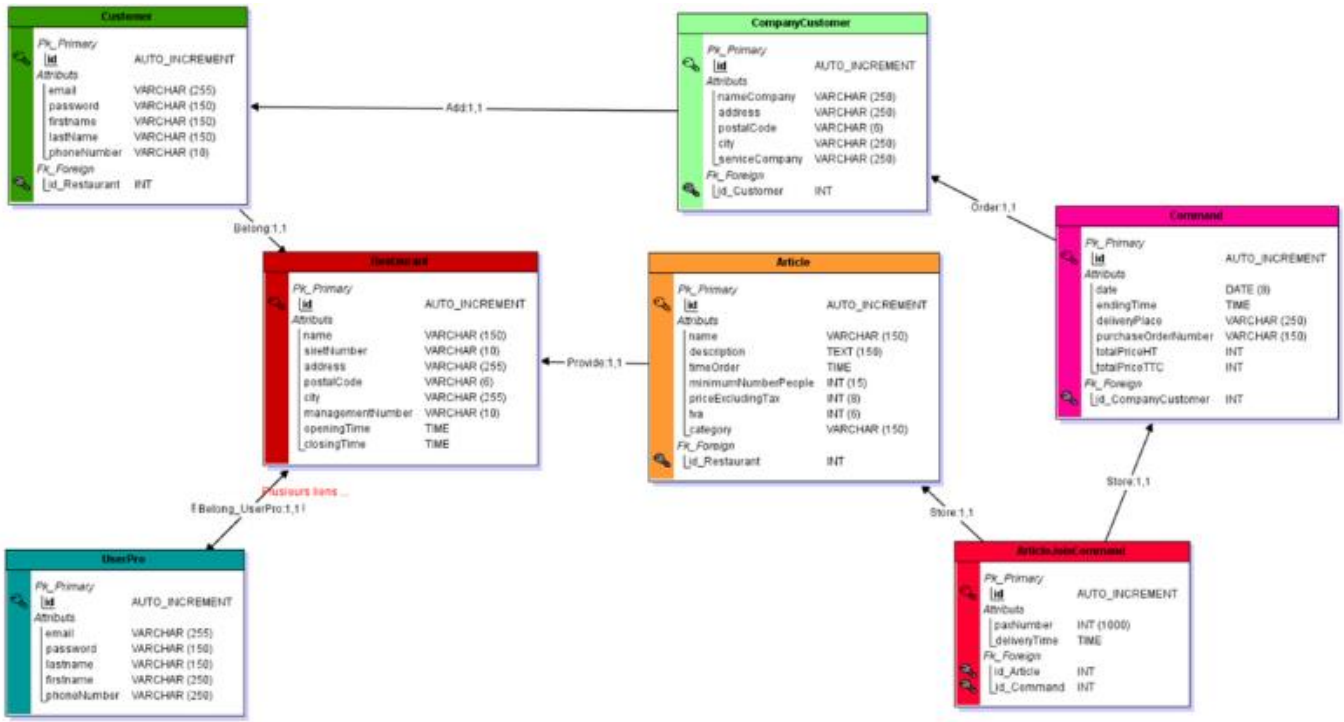
Terminologie technique associée :

- **Entité (MCD) :** Un objet ou concept distinct (par exemple, un client, un produit).
- **Relation (MCD) :** Lien entre deux entités, indiquant une association (par exemple, un client "achète" un produit).

14. Transformation du MCD en Modèle Logique de Données (MLD)

Le MLD va plus loin dans la spécification des données en définissant les tables, les colonnes, les clés primaires, les clés étrangères, etc. C'est une étape vers la mise en œuvre technique de la base de données.

Il a été créé pour donner une structure concrète à la manière dont les données seront stockées dans une base de données. Le MLD traduit les concepts abstraits du MCD en structures de données réelles.



Terminologie technique associée :

- **Table (MLD) :** Structure qui stocke les données de manière organisée, chaque ligne représentant une entrée distincte.
- **Colonne (MLD) :** Champ spécifique à une propriété dans une table.
- **Clé primaire (MLD) :** Colonne ou groupe de colonnes qui identifie de manière unique chaque enregistrement dans une table.
- **Clé étrangère (MLD) :** Colonne qui établit une relation avec la clé primaire d'une autre table.

15. Relations entre les Tables

Les relations entre les tables sont cruciales dans la conception de bases de données relationnelles. Elles définissent comment les données sont liées les unes aux autres et permettent d'établir des connexions significatives. Voici quelques concepts importants à comprendre concernant les relations entre les tables :

Types de Relations

- **1:1 (One-to-One)**: Chaque enregistrement d'une table est associé à un seul enregistrement dans une autre table, et vice versa. Cette relation est représentée par des clés étrangères.

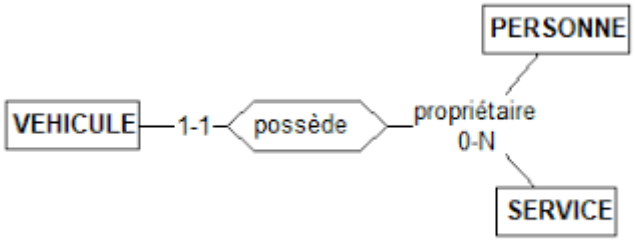
- **1:N (One-to-Many)**: Chaque enregistrement dans une table peut être associé à plusieurs enregistrements dans une autre table, mais chaque enregistrement dans l'autre table est associé à un seul enregistrement dans la première table.

- **N:M (Many-to-Many)**: Chaque enregistrement dans une table peut être associé à plusieurs enregistrements dans une autre table, et vice versa. Cette relation nécessite généralement une table de jointure pour gérer les correspondances.

Clés dans les Relations

- Clé Primaire : Une colonne ou un groupe de colonnes qui identifie de manière unique chaque enregistrement dans une table. Souvent utilisée pour établir des relations avec d'autres tables.

- Clé Étrangère : Une colonne qui établit une relation avec la clé primaire d'une autre table. Elle est utilisée pour créer des liens entre les données dans différentes tables.

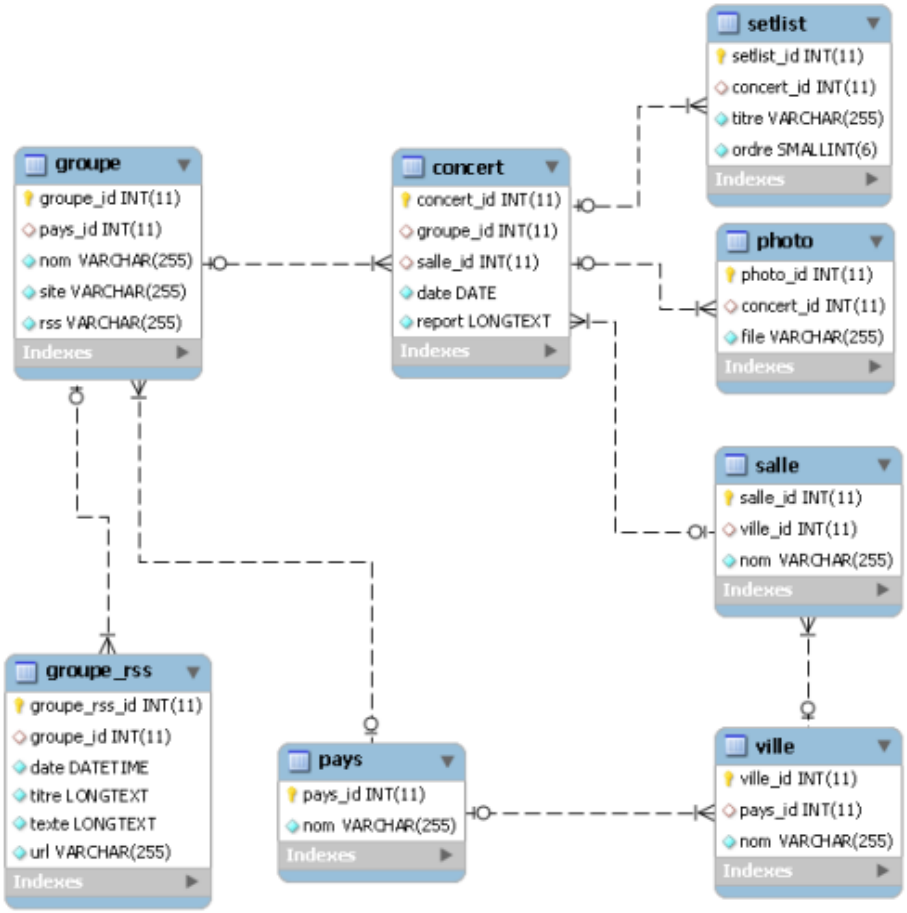


16. Types de Données pour les Entités

Les entités dans une base de données peuvent représenter différents types d'informations. Comprendre les types de données est crucial pour garantir la cohérence et l'intégrité des données. Voici ce qu'il faut savoir :

Types de Données Courants :

- VARCHAR(N) : Chaîne de caractères variable avec une longueur maximale de N caractères.
- INT : Nombre entier.
- DATE : Date au format YYYY-MM-DD.
- FLOAT : Nombre à virgule flottante.
- BOOL : Valeur booléenne (VRAI ou FAUX).



17. NoSQL (Not Only SQL)

NoSQL est un terme générique qui englobe une variété de technologies de gestion de bases de données qui ne suivent pas le modèle relationnel traditionnel SQL. Contrairement aux bases de données SQL, les bases de données NoSQL sont conçues pour gérer des types de données variés et offrir une évolutivité horizontale.

Principales caractéristiques du NoSQL :

- Modèle de données flexible : Les bases de données NoSQL sont souvent basées sur des modèles de données flexibles tels que les documents, les paires clé-valeur, les graphiques, ou les familles de colonnes. Cela permet de stocker différents types de données de manière efficace.

- Évolutivité horizontale : Les bases de données NoSQL sont conçues pour évoluer facilement en ajoutant de nouveaux serveurs au lieu d'augmenter la puissance d'un serveur existant (évolutivité verticale).

- Haute performance : Les bases de données NoSQL sont optimisées pour des opérations spécifiques, ce qui peut offrir des performances élevées dans des cas d'utilisation particuliers.

- Pas de schéma fixe : Contrairement aux bases de données relationnelles qui nécessitent un schéma fixe, les bases de données NoSQL peuvent gérer des données sans schéma préalable.

Types de bases de données NoSQL courants :

- Bases de données de documents : Stockent des données sous forme de documents (MongoDB, CouchDB).

- Bases de données de paires clé-valeur : Stockent des données sous forme de paires clé-valeur (Redis, DynamoDB).

- Bases de données de colonnes : Stockent des données dans des colonnes plutôt que des lignes (Cassandra, HBase).

- Bases de données graphiques : Optimal pour stocker et interroger des données graphiques (Neo4j, ArangoDB).