

Bases de données NoSQL : Cassandra

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

`elmouelhi.achref@gmail.com`



cassandra

- 1 Introduction
- 2 Mise en place
 - Installation locale
 - Cloud
- 3 Particularités de Cassandra
- 4 Gestion de KeySpace
 - Création
 - Suppression
 - Consultation
 - Utilisation

5 Gestion de tables

- Création
- Consultation
- Suppression

6 Gestion de colonnes

- Ajout d'une colonne
- Suppression d'une colonne
- Renommer une clé primaire

7 Insertion

8 Suppression

- DELETE FROM
- TRUNCATE

9 Modification

10 Batch

11 Clé de partition et clé de clustering

12 Lecture de données

- SELECT ... FROM ...
- LIMIT
- WHERE
- IN

13 Cas d'un tableau

- Création
- Insertion
- Sélection
- Modification

Cassandra

Cassandra ou Apache Cassandra

- Système de gestion de base de données faisant partie de la mouvance **NoSQL**
- Créé en 2008 par **Facebook** puis maintenu par la fondation **Apache** depuis 2009
- Orienté colonne ou wide-column en anglais
- Décrit par **Facebook** comme un SGBD **Big Table**
- Open-source
- Développé principalement en **Java**
- Utilisant un langage de requêtes **CQL** **Cassandra Query Language**)

Cassandra

Cassandra : histoire

- En 2009, **Facebook** publie un article parlant de la création d'une application interne décentralisée, appelée **Cassandra**, pour stocker les données
- Lien vers l'article : <https://docs.datastax.com/en/articles/cassandra/cassandrathenandnow.html>
- **Facebook** avait besoin de stocker les messages échangés : environ 135 000 000 000 de messages par mois.
- Aucune base de données relationnelle permettait de les stocker
- **Facebook** a testé en parallèle **Cassandra** et **HBase** et a fini par garder **HBase** pour sa stabilité.
- **Cassandra** a été confié à **Apache** et est devenu open-source.
- En 2010, des ingénieurs de **Facebook** ont créé **DataStax** : qui étendent les fonctionnalités de **Cassandra** avec des fonctionnalités supplémentaires telles que la sécurité, la surveillance, et la gestion des performances

Cassandra

CQL : Similitudes avec SQL

- **Syntaxe de base** : `SELECT`, `INSERT`, `UPDATE` et `DELETE`
- **Types de données** : **Cassandra** supporte les chaînes de caractères, les entiers, les flottants, les booléens...
- **Clés primaires** : **Cassandra** utilise des clés primaires pour identifier de manière unique les lignes dans une table.

Cassandra

CQL : Différences par rapport à SQL

- **Modèle de données : Cassandra** est basé sur des colonnes et des familles de colonnes, plutôt que sur des tables.
- **Absence de jointure et de clé étrangère** : les schémas de données doivent être conçus en fonction des requêtes et des modèles de lecture.
- **Pas de transactions ACID** : **Cassandra** privilégie la disponibilité et la tolérance aux partitions.
- **Répartition et réplication des données** : **Cassandra** privilégie la disponibilité et la tolérance aux partitions.
- ...

Cassandra

Cassandra : Quelques chiffres [DB-Engines]

- SGBD NoSQL orienté colonne le plus utilisé en 2024, 2023, 2022...
- Classé onzième dans le classement des SGBD **SQL** et **NoSQL**
- Classé quatrième dans le classement des SGBD **NoSQL**
- ...

Cassandra

MongoDB : utilisé par

- **Twitter**
- **Netflix**
- **Spotify**
- ...

Cassandra

Pour utiliser **Cassandra** : deux options principales

- **Installation locale** : télécharger et installer **Cassandra**
- **DataStax** : opter pour une version gérée dans le cloud

Cassandra

Pré-installations

- **Java 8** pour **Cassandra 3.***
- **Java 8** ou **Java 11** pour **Cassandra 4.*** et **Cassandra 5.***
- **Python 3.6+** pour **CQLSH**

Cassandra

Première solution : installation locale

- Allez à `https://cassandra.apache.org/doc/stable/cassandra/getting\_started/installing.html`
- Choisissez le mode d'installation et suivez les instructions

Cassandra

Deuxième solution : cloud

- Allez à `https://www.datastax.com/`
- Cliquez sur Try For Free
- Créez un compte
- Créez une base de données
- Cliquez sur la base de données, allez dans Data Explorer et créez le namespace (Key Space) `mon_keyspace`
- Cliquez sur **CQL Console**

Cassandra

Cassandra vs SQL

- Base = Base
- KeySpace : { sous base } (pas de correspondance en **SQL**)
- Table = Collection
- Enregistrement (tuple) = tuple
- Clé primaire = clé primaire
- Pas de clé étrangère dans **Cassandra**

Cassandra

Lister les commandes possibles sur les bases de données avec CQL

HELP

Cassandra

MongoDB : types de données

- int, bigint, smallint, tinyint
- boolean
- float, double, decimal
- text, varchar
- uuid
- date, time, timestamp
- list, set, map
- ...

Cassandra

Pour créer un KeySpace

```
create_keyspace_statement ::= CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name  
                             WITH options
```

© Achref EL MOUELHI ©

Cassandra

Pour créer un KeySpace

```
create_keyspace_statement ::= CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name  
                             WITH options
```

Exemple

```
CREATE KEYSPACE IF NOT EXISTS mon_keyspace  
WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
```

Cassandra

Pour créer un Keyspace

```
create_keyspace_statement ::= CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name  
                           WITH options
```

Exemple

```
CREATE KEYSPACE IF NOT EXISTS mon_keyspace  
WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
```

Explication

- `replication` : un map qui définit la stratégie de réplication pour le keyspace
- `class` : spécifie la stratégie de réplication. Les valeurs communes incluent `SimpleStrategy` pour un environnement de développement ou de test (ou pour un cluster en un seul datacenter) et `NetworkTopologyStrategy` pour les environnements de production ou les clusters répartis sur plusieurs datacenters.
- `replication_factor` (utilisé avec `SimpleStrategy`) détermine combien de copies de chaque donnée seront stockées dans le cluster. Pour `NetworkTopologyStrategy`, vous spécifierez le facteur de réplication par datacenter.

Cassandra

Pour supprimer un keyspace

```
DROP KEYSPACE nom_keyspace;
```

© Achref EL MOUL

Cassandra

Pour supprimer un keyspace

```
DROP KEYSPACE nom_keyspace;
```

pour éviter le message d'erreur si le keyspace n'existe pas

```
DROP KEYSPACE [IF EXISTS] nom_keyspace;
```

Cassandra

Pour afficher tous les keyspace

```
DESC KEYSPACES;
```

Cassandra

Pour utiliser le keyspace

```
USE nom_keyspace;
```

Cassandra

Création

```
CREATE TABLE nom_table (  
    nom_colonne1 type,  
    ...  
    nom_colonneN type,  
  
    [PRIMARY KEY (colonnes_cles_principales)]  
);
```

© Achref EL

Cassandra

Création

```
CREATE TABLE nom_table (  
    nom_colonne1 type,  
    ...  
    nom_colonneN type,  
  
    [PRIMARY KEY (colonnes_cles_principales)]  
);
```

Dans le cas d'une clé primaire mono-colonne

```
CREATE TABLE nom_table (  
    nom_colonne1 type PRIMARY KEY,  
    ...  
    nom_colonneN type  
);
```

Cassandra

Création

```
CREATE TABLE nom_table (  
    nom_colonne1 type,  
    ...  
    nom_colonneN type,  
  
    [PRIMARY KEY (colonnes_cles_principales)]  
) [ WITH table_options ];
```

© Achref EL

Cassandra

Création

```
CREATE TABLE nom_table (  
    nom_colonne1 type,  
    ...  
    nom_colonnen type,  
  
    [PRIMARY KEY (colonnes_cles_principales)]  
)  
[ WITH table_options ];
```

Dans le cas d'une clé primaire mono-colonne

```
CREATE TABLE nom_table (  
    nom_colonne1 type PRIMARY KEY,  
    ...  
    nom_colonnen type  
)  
[ WITH table_options ];
```

Cassandra

Création

```
CREATE TABLE personnes (  
    id uuid,  
    nom text,  
    prenom text,  
    PRIMARY KEY(id)  
) WITH comment='Ces données ne sont pas confidentielles';
```

© Achref EL M...

Cassandra

Création

```
CREATE TABLE personnes (  
    id uuid,  
    nom text,  
    prenom text,  
    PRIMARY KEY(id)  
) WITH comment='Ces données ne sont pas confidentielles';
```

Dans le cas d'une clé primaire mono-colonne

```
CREATE TABLE persons (  
    id int PRIMARY KEY,  
    nom text,  
    prenom text  
);
```

Cassandra

Pour afficher la liste des tables

```
DESCRIBE TABLES;
```

© Achref EL MOUELHI ©

Cassandra

Pour afficher la liste des tables

```
DESCRIBE TABLES;
```

Ou

```
DESC TABLES;
```

Cassandra

Pour afficher la liste des tables

```
DESCRIBE TABLES;
```

Ou

```
DESC TABLES;
```

Dans le cas d'une clé primaire mono-colonne

```
personnes persons
```

Cassandra

Pour consulter le script ayant permis de créer la table

```
DESCRIBE nom_table;
```

© Achref EL MOUL

Cassandra

Pour consulter le script ayant permis de créer la table

```
DESCRIBE nom_table;
```

Ou

```
DESC nom_table;
```

Cassandra

Pour supprimer une table

```
DROP TABLE nom_table;
```

© Achref EL MOUËL

Cassandra

Pour supprimer une table

```
DROP TABLE nom_table;
```

TRUNCATE supprime toutes les données d'une table mais pas la table

```
TRUNCATE [TABLE] nom_table;
```

Cassandra

Ajouter une colonne (qui sera la dernière dans la table)

```
ALTER TABLE nom_table  
ADD nom_colonne type;
```

Cassandra

Pour supprimer une colonne

```
ALTER TABLE personnes  
DROP age;
```

Cassandra

Renommer une clé primaire

```
ALTER TABLE personnes  
RENAME ancien_nom TO nouveau_nom;
```

© Achref EL MOUELHI ©

Cassandra

Renommer une clé primaire

```
ALTER TABLE personnes  
RENAME ancien_nom TO nouveau_nom;
```

Renommer clé primaire composée

```
ALTER TABLE nom_table  
RENAME  
ancien_nom1 TO nouveau_nom1 AND  
...  
ancien_nomN TO nouveau_nomN;
```

Cassandra

Renommer une clé primaire

```
ALTER TABLE personnes  
RENAME ancien_nom TO nouveau_nom;
```

Renommer clé primaire composée

```
ALTER TABLE nom_table  
RENAME  
ancien_nom1 TO nouveau_nom1 AND  
...  
ancien_nomN TO nouveau_nomN;
```

Remarque

Une colonne non clé primaire ne peut pas être renommée ?

Cassandra

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES (colonnes) (  
    valeur_colonne1, ..., valeur_colonneN);
```

Cassandra

Insérer une valeur pour chaque colonne

```
INSERT INTO nom_table VALUES (colonnes) (  
    valeur_colonne1, ..., valeur_colonneN);
```

Le SGBD affectera les valeurs aux colonnes dans l'ordre.

Cassandra

Exemple

```
INSERT INTO persons (id, nom, prenom) VALUES (1, 'Cooper', 'David');
```

© Achref EL MOUËL

Cassandra

Exemple

```
INSERT INTO persons (id, nom, prenom) VALUES (1, 'Cooper', 'David');
```

Exemple avec uuid

```
INSERT INTO personnes (id, nom, prenom) VALUES (uuid(), 'Cooper', 'David');
```

Cassandra

Si l'identifiant existe, alors pas d'erreur mais une modification sera effectuée

```
INSERT INTO persons (id, nom, prenom) VALUES (1, 'Doe', 'John');
```

© Achref EL MOUELHI ©

Cassandra

Si l'identifiant existe, alors pas d'erreur mais une modification sera effectuée

```
INSERT INTO persons (id, nom, prenom) VALUES (1, 'Doe', 'John');
```

Cette requête génère une erreur car la présence la clé est obligatoire

```
INSERT INTO persons (nom, prenom) VALUES ('Dalton', 'Jack');
```

Cassandra

Si l'identifiant existe, alors pas d'erreur mais une modification sera effectuée

```
INSERT INTO persons (id, nom, prenom) VALUES (1, 'Doe', 'John');
```

Cette requête génère une erreur car la présence la clé est obligatoire

```
INSERT INTO persons (nom, prenom) VALUES ('Dalton', 'Jack');
```

Les valeurs pour les autres colonnes peuvent ne pas être spécifiées

```
INSERT INTO persons (id, nom) VALUES (2, 'Dalton');
```

Cassandra

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table  
[WHERE conditions];
```

© Achref EL MOUELHI ©

Cassandra

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table  
[WHERE conditions];
```

Exemple

```
DELETE FROM personnes  
WHERE salaire > 2000 AND ville = 'Marseille';
```

Cassandra

Supprimer des tuples respectant une ou plusieurs conditions

```
DELETE FROM nom_table  
[WHERE conditions];
```

Exemple

```
DELETE FROM personnes  
WHERE salaire > 2000 AND ville = 'Marseille';
```

Ou

```
DELETE FROM personnes  
WHERE ville in ('Marseille', 'Paris');
```

Cassandra

Pour vider un champ (le remplacer par `null`)

```
DELETE nom  
FROM nom_table  
[WHERE condtions];
```

Cassandra

Supprimer tous les tuples d'une table

```
DELETE FROM nom_table;
```

© Achref EL MOUELHI

Cassandra

Supprimer tous les tuples d'une table

```
DELETE FROM nom_table;
```

Remarque

Cette requête supprime toutes les données de la table, mais pas la table. Donc, le résultat sera une table vide.

Cassandra

TRUNCATE aussi permet de supprimer toutes les données d'une table

```
TRUNCATE nom_table;
```

© Achref EL MOUELHI ©

Cassandra

TRUNCATE aussi permet de supprimer toutes les données d'une table

```
TRUNCATE nom_table;
```

Remarques

- **TRUNCATE** fonctionne comme un **DELETE** sans **WHERE**.
- **TRUNCATE** est plus rapide que **DELETE** car elle supprime puis recrée la table.
- **DELETE** supprime les tuples un par un.
- **TRUNCATE** ne prend pas de clause **WHERE**.

Cassandra

Modifier des tuples respectant une ou plusieurs conditions

```
UPDATE nom_table  
SET nom_colonne1 = valeur1  
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]  
WHERE condition;
```

© Achref EL MOU

Cassandra

Modifier des tuples respectant une ou plusieurs conditions

```
UPDATE nom_table  
SET nom_colonne1 = valeur1  
[nom_colonne2 = valeur2, ..., nom_colonneN = valeurN]  
WHERE condition;
```

Exemple

```
UPDATE personnes  
SET salaire = 1600, ville = 'Toulouse'  
WHERE nom = 'benatia';
```

Cassandra

Modifier tous les tuples

```
UPDATE enseignant  
SET ville = 'Marseille' ;
```

Cassandra

Pour minimiser le nombre de va et vient entre le client et le serveur, on peut regrouper plusieurs instructions d'écritures dans un batch

```
BEGIN BATCH
  INSERT INTO persons (id, nom, prenom) VALUES (3, 'Linus', 'Benjamin'
);
  UPDATE persons SET nom = 'Shephard' WHERE id = 2;
  INSERT INTO persons (id, nom, prenom) VALUES (4, 'Lock', 'John');
  DELETE nom FROM persons WHERE id = 1;
APPLY BATCH;
```

Cassandra

Clé primaire : rappel

- **Clé primaire simple** : colonne unique utilisée pour identifier de manière unique chaque enregistrement dans la table
- **Clé primaire composite** (également connue sous le nom de clé primaire composée ou multiple) : clé primaire qui utilise plus d'une colonne pour identifier de manière unique chaque ligne dans une table $\Rightarrow$ la combinaison des valeurs dans ces colonnes doit être unique dans la table.

Cassandra

Définitions

- **nœud**

- unité autonome de calcul et de stockage
- serveur unique ou une instance de la base de données qui stocke une partie des données et qui participe au traitement des requêtes
- pouvant fonctionner sur une machine physique ou virtuelle

- **Cluster**

- { nœud }
- travaillant ensemble pour se comporter comme un système unique
- conçu pour permettre une scalabilité horizontale : la capacité du système peut être augmentée en ajoutant simplement plus de nœuds au cluster

Cassandra

Clé de partition

- Utilisée pour distribuer les données sur les nœuds du cluster : les données sont réparties en fonction de la valeur de la clé de partition.
- Garantissant que toutes les lignes partageant la même valeur de clé de partition se trouvent sur le même nœud.

© Achrel

Cassandra

Clé de partition

- Utilisée pour distribuer les données sur les nœuds du cluster : les données sont réparties en fonction de la valeur de la clé de partition.
- Garantissant que toutes les lignes partageant la même valeur de clé de partition se trouvent sur le même nœud.

Remarque

Sur un nœud donné, on peut stocker des données avec différentes clés de partition.

Cassandra

Clé de clustering

- { colonnes d'une clé primaire composite non-utilisées dans la clé de partition }.
- Utilisée pour trier les données à l'intérieur d'une partition.

© Achref EL M

Cassandra

Clé de clustering

- { colonnes d'une clé primaire composite non-utilisées dans la clé de partition }.
- Utilisée pour trier les données à l'intérieur d'une partition.

Remarque

Une clé de partition ne peut pas être composée.

Cassandra

Clé de clustering, clé de partition et clé primaire

- **Avec une clé primaire simple**

- La colonne de la clé primaire définit la clé de partition.
- Il n'y a pas de clé de clustering.

- **Avec une clé primaire composite**

- La première partie de la clé composite (celle de gauche) définit la clé de partition.
- Les parties suivantes définissent les clés de clustering et permettant donc le tri des données au sein de la partition.

Cassandra

Considérons le script suivant pour la création des tables `personnes` et `vehicules`

```
CREATE TABLE personnes (  
    num int PRIMARY KEY,  
    nom text,  
    prenom text,  
    salaire int,  
    ville text  
);  
  
CREATE TABLE vehicules (  
    nump int,  
    immatriculation int,  
    marque text,  
    modele text,  
    annee int,  
    PRIMARY KEY (nump, immatriculation)  
);
```

Cassandra

Considérons le script suivant pour la création des tables `personnes` et `vehicules`

```
CREATE TABLE personnes (  
    num int PRIMARY KEY,  
    nom text,  
    prenom text,  
    salaire int,  
    ville text  
);  
  
CREATE TABLE vehicules (  
    nump int,  
    immatriculation int,  
    marque text,  
    modele text,  
    annee int,  
    PRIMARY KEY (nump, immatriculation)  
);
```

Remarque

Pour la table `vehicules`, `nump` est la clé de partition et `immatriculation` est la clé de clustering.

Cassandra

Script d'insertion de données

```
INSERT INTO personnes (num, nom, prenom, salaire, ville) VALUES
(1, 'Cohen', 'Sophie', 2000, 'Marseille');
INSERT INTO personnes (num, nom, prenom, salaire, ville) VALUES
(2, 'Leberre', 'Bernard', 1500, 'Marseille');
INSERT INTO personnes (num, nom, prenom, salaire, ville) VALUES
(3, 'Benamar', 'Pierre', 1800, 'Lyon');
INSERT INTO personnes (num, nom, prenom, salaire, ville) VALUES
(4, 'Hadam', 'Karim', 2500, 'Paris');
INSERT INTO personnes (num, nom, prenom, salaire, ville) VALUES
(5, 'Wick', 'John', 3000, 'Paris');
```

```
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(100, 'Peugeot', '5008', 2018, 5);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(200, 'Renault', 'clio', 2000, 4);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(300, 'Ford', 'fiesta', 2010, 1);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(400, 'Peugeot', '106', 2002, 3);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(500, 'Citroen', 'C4', 2015, 4);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(600, 'Ford', 'Kuga', 2019, 6);
INSERT INTO vehicules (immatriculation, marque, modele, annee, nump) VALUES
(700, 'Fiat', 'punto', 2008, 5);
```

Cassandra

Remarque

Une requête **CQL** de lecture est composée de deux clauses obligatoires : `SELECT` et `FROM`.

© Achref EL MOUELHANI

Cassandra

Remarque

Une requête **CQL** de lecture est composée de deux clauses obligatoires : **SELECT** et **FROM**.

Structure d'une requête SQL

```
SELECT nom_colonne(s)
FROM nom_table(s)
WHERE condition(s)
GROUP BY nom_colonne(s)
HAVING condition(s)
ORDER BY nom_colonne(s)
LIMIT nombre;
```

Cassandra

Pour sélectionner toutes les données de la table `personne`

```
SELECT *  
FROM personnes;
```

© Achref EL MOUELHI ©

Cassandra

Pour sélectionner toutes les données de la table `personne`

```
SELECT *  
FROM personnes;
```

Le résultat

num	nom	prenom	salaire	ville
5	Wick	John	3000	Paris
1	Cohen	Sophie	2000	Marseille
2	Leberre	Bernard	1500	Marseille
4	Hadad	Karim	2500	Paris
3	Benamar	Pierre	1800	Lyon

Cassandra

Et si on voulait sélectionner que les deux premières personnes

```
SELECT *  
FROM personnes  
LIMIT 2;
```

© Achref EL MOUL

Cassandra

Et si on voulait sélectionner que les deux premières personnes

```
SELECT *  
FROM personnes  
LIMIT 2;
```

Le résultat

num	nom	prenom	salaire	ville
5	Wick	John	3000	Paris
1	Cohen	Sophie	2000	Marseille

Cassandra

Pour filtrer les colonnes (ici la ville de chaque personne)

```
SELECT ville  
FROM personnes;
```

Le résultat

ville

Paris
Marseille
Marseille
Paris
Lyon

Cassandra

Pour filtrer les colonnes (ici la ville de chaque personne)

```
SELECT ville  
FROM personnes;
```

Le résultat

ville

Paris
Marseille
Marseille
Paris
Lyon

Comment on fait pour supprimer les doublons ?

Cassandra

Pour sélectionner les personnes dont la ville = 'Marseille'

```
SELECT *  
FROM personnes  
WHERE ville = 'Marseille'  
ALLOW FILTERING;
```

Le résultat

num	nom	prenom	salaire	ville
1	Cohen	Sophie	2000	Marseille
2	Leberre	Bernard	1500	Marseille

Cassandra

Il est possible de définir plusieurs conditions en utilisant les opérateurs logiques

- AND
- OR
- !

Cassandra

Exercice 1

Écrire une requête **SQL** qui permet de sélectionner les personnes qui habitent Marseille ou Lyon.

© Achref EL MOU

Cassandra

Exercice 1

Écrire une requête **SQL** qui permet de sélectionner les personnes qui habitent Marseille ou Lyon.

Exercice 2

Écrire une requête **SQL** qui permet de sélectionner les personnes dont le salaire est compris entre 2000 et 3000.

Cassandra

Pour sélectionner les personnes ayant un salaire = 2000 ou 3000

```
SELECT *  
FROM personnes  
WHERE salaire IN (2000, 3000)  
ALLOW FILTERING;
```

Le résultat

num	nom	prenom	salaire	ville
5	Wick	John	3000	Paris
1	Cohen	Sophie	2000	Marseille

Cassandra

Exercice 3

Écrire une requête **SQL** qui permet de sélectionner les personnes qui habitent Marseille et dont le salaire est soit inférieur à 2000 soit supérieur à 2500.

Cassandra

Considérons la table `etudiants` contenant une colonne `notes` de type liste de réels

```
CREATE TABLE étudiants (  
    id int PRIMARY KEY,  
    nom text,  
    prenom text,  
    notes list<float>  
);
```

Cassandra

Pour insérer quelques étudiants avec une liste de notes

```
INSERT INTO etudiants (id, nom, prenom, notes) VALUES (1, 'Doe', 'John', [10, 12, 14]);  
INSERT INTO etudiants (id, nom, prenom, notes) VALUES (2, 'Dalton', 'Jack', [20, 15, 10]);  
INSERT INTO etudiants (id, nom, prenom, notes) VALUES (3, 'White', 'Sophie', [10, 8, 4]);
```

Cassandra

Pour sélectionner selon une valeur de la liste

```
SELECT * FROM etudiants WHERE notes CONTAINS 12;
```

© Achref EL MOUËZ

Cassandra

Pour sélectionner selon une valeur de la liste

```
SELECT * FROM etudiants WHERE notes CONTAINS 12;
```

En cas de message d'erreur par rapport à la performance

```
SELECT * FROM etudiants WHERE notes CONTAINS 12 ALLOW FILTERING;
```

Cassandra

Pour ajouter une note

```
UPDATE etudiants SET notes = notes + [15] WHERE id = 2;
```

© Achref EL MOUELHI ©

Cassandra

Pour ajouter une note

```
UPDATE etudiants SET notes = notes + [15] WHERE id = 2;
```

Remarque

Ajouter un élément dans un tableau selon son indice, le modifier ou le supprimer n'est plus autorisé par **Cassandra** pour son coût. En effet, cela nécessite une lecture puis une réécriture de la partie modifiée de la liste.

Cassandra

Pour ajouter une note

```
UPDATE etudiants SET notes = notes + [15] WHERE id = 2;
```

Remarque

Ajouter un élément dans un tableau selon son indice, le modifier ou le supprimer n'est plus autorisé par **Cassandra** pour son coût. En effet, cela nécessite une lecture puis une réécriture de la partie modifiée de la liste.

La meilleure solution consiste à réaffecter la liste

```
UPDATE etudiants SET notes = [15, 10, 13] WHERE id = 2;
```