**Rapport de Migration vers un Modèle Graphe avec Neo4j**

**Préparé par : Mohamed Diallo**

**2024-2025**

**Yanis Rebahi et Dung Vu**

**Sommaire**

1. Introduction
2. Analyse du Modèle Relationnel
3. Proposition du Modèle Graphe
4. Migration des Données

4.1 Exploration et Préparation des Données

4.2 Script Python de Migration

4.3 Insertion des Données dans Neo4j

1. Validation et Comparaison des Résultats

5.1 Exemples de Requêtes Cypher et Résultats

5.2 Analyse des Performances

1. Conclusion et Recommandations
2. Annexes

**Introduction**

Dans le cadre de ce projet, nous avons entrepris la migration des données relatives aux crimes et délits en France (2012-2022) depuis une base relationnelle vers une base orientée graphe avec Neo4j.

Cette démarche répond à des besoins croissants d’analyse des relations complexes entre entités telles que les infractions, les départements, les forces de l’ordre, et les faits criminels. Le modèle relationnel classique montre ses limites en termes de performances et de flexibilité dès lors que l’on aborde des analyses relationnelles complexes. Les bases relationnelles sont bien adaptées aux **données structurées** et aux **requêtes transactionnelles classiques**, mais elles présentent des limites significatives dès que l'on s'intéresse à **l'analyse des relations** entre les entités stockées.

Dans une base relationnelle, les entités sont stockées sous forme de **tables distinctes**, reliées par des **clés étrangères**. Lorsqu'une analyse complexe nécessite de récupérer des informations connectées à plusieurs niveaux, cela implique des **jointures multiples** qui deviennent rapidement inefficaces sur de grands volumes de données.

Également, les bases relationnelles ne sont pas conçues pour **explorer des relations multiples sur plusieurs niveaux**. Par exemple, si l’on souhaite analyser **l’évolution des types de crimes en fonction des interactions entre plusieurs départements voisins**, il est nécessaire d'écrire des requêtes récursives compliquées. La rigidité du modèle relationnel (ici SQL) est aussi une limite. Si nous devons ajouter de nouvelles relations entre entités (par exemple, relier un crime similaire à un autre ou modéliser des réseaux de groupes criminels dans les départements) nécessite de modifier la structure de base de la table, voire de la base.

Neo4j, en tant que base de données orientée graphe, offre une solution plus adaptée pour répondre à ces problématiques. Il repose sur une **modélisation flexible** où les relations sont **des objets de premier ordre** plutôt que des contraintes gérées par des clés étrangères. Le modèle excelle dans la navigation des graphes, ce qui permet d’explorer les connexions multiples sur plusieurs niveaux sans récursion SQL.

Ce rapport détaille les différentes phases du projet, de l’analyse du modèle relationnel à la conception du modèle graphe, en passant par la migration des données et la validation des résultats obtenus.

**Analyse du modèle relationnel**

Le modèle relationnel initial repose sur plusieurs entités clés, chacune ayant un rôle précis dans l’organisation des données. Tout d'abord, l'entité **INFRACTION** représente les types de crimes ou délits, comme le vol ou l’homicide, permettant de classer les infractions. Ensuite, l'entité **FORCE\_ORDRE** désigne l’organisme chargé de constater l’infraction, tel que la Police Nationale ou la Gendarmerie Nationale, ce qui lie les infractions à l'autorité responsable. L'entité **DÉPARTEMENT** sert à localiser géographiquement chaque infraction, ce qui permet d’associer chaque crime à une zone précise. De manière optionnelle, l’entité **RÉGION** regroupe plusieurs départements, facilitant ainsi l’analyse à un niveau géographique plus large. Enfin, l'entité **FAIT** enregistrée les statistiques relatives aux infractions, en indiquant le nombre de faits par année, département et force de l’ordre, créant une vue temporelle et géographique des données.

Le modèle relationnel présente plusieurs avantages : il offre une structure claire, avec des contraintes bien définies sur les données (par exemple, des clés primaires et étrangères), assurant l'intégrité des informations. Il est également robuste, car il repose sur une modélisation normalisée qui évite la redondance des données, ce qui permet une gestion efficace des informations. De plus, il est bien adapté aux requêtes analytiques classiques, permettant d'effectuer facilement des calculs ou des agrégations simples sur les données.

Cependant, ce modèle présente également certaines limitations. Lorsqu'il s'agit de traiter de gros volumes de données, les performances peuvent être réduites, en particulier lors de l’exécution de jointures complexes entre les différentes tables. De plus, la gestion des relations indirectes (par exemple, celles entre des infractions ou des réseaux criminels) devient de plus en plus complexe et coûteuse en ressources.

Enfin, le modèle relationnel manque de flexibilité lorsqu'il s'agit d'ajouter de nouvelles entités ou relations, car chaque modification nécessite une révision de la structure de la base de données, ce qui peut rendre l’adaptation difficile .

**Proposition d’un modèle graphe**

Pour surmonter les limitations du modèle relationnel, nous avons opté pour un modèle basé sur Neo4j, une base de données orientée graphe. Ce choix permet de tirer parti des avantages inhérents à la structure de graphe, notamment dans le cadre de l'analyse des données criminelles. En effet, ce modèle facilite l'exploration des relations directes et indirectes entre les crimes, les forces de l’ordre et les localisations géographiques, offrant ainsi une vision plus complète et détaillée des interactions entre ces différentes entités. Il permet également de mieux identifier et analyser des réseaux criminels complexes, tout en facilitant la détection des récidives. De plus, le modèle graphe offre une gestion efficace des connexions multiples entre entités, sans les contraintes des jointures coûteuses qui ralentissent les requêtes dans les bases relationnelles.

Le modèle graphe se compose principalement de nœuds et de relations entre ces nœuds. Les nœuds représentent les entités principales, telles qu’**INFRACTION**, **FORCE\_ORDRE**, **FAIT**, **DÉPARTEMENT**, et **BRIGADE**, tandis que les relations définissent comment ces entités sont connectées. Par exemple, une relation [:CONSTATE\_PAR] lie une **INFRACTION** à une **FORCE\_ORDRE**, indiquant ainsi que l'infraction a été constatée par une force de l'ordre. De même, la relation [:OCCURE\_DANS] relie une **INFRACTION** à un **DÉPARTEMENT**, permettant de situer géographiquement chaque crime. Les FAITS, qui enregistrent les infractions, sont eux-mêmes reliés à des INFRACTIONS par la relation [:CONCERNE] et sont enregistrés par une BRIGADE via la relation [:ENREGISTRE\_PAR]. Enfin, des relations supplémentaires comme [:COUVRE] permettent de lier une BRIGADE à un PÉRIMÈTRE, offrant une vue d'ensemble sur les zones de couverture des forces de l'ordre. Ce modèle permet ainsi une flexibilité maximale pour explorer les données et découvrir des liens complexes, tout en garantissant une meilleure performance dans l'analyse.

**Migration des Données**

***4.1 Exploration et Préparation des Données***

Dans un premier temps, nous avons procédé à une **exploration approfondie des données** afin d’identifier les ajustements nécessaires avant leur migration vers le modèle graphe. Cette analyse a révélé une **disparité structurelle majeure** entre les données de la **Police Nationale (PN)** et celles de la **Gendarmerie Nationale (GN)** : la **PN est organisée en périmètres (CSP – Commissariats de Sécurité Publique), tandis que la GN ne dispose pas de cette structure**, étant organisée au niveau des départements et brigades.

Ce déséquilibre posait un **problème d’uniformisation** dans le modèle graphe, car les **relations entre forces de l’ordre et infractions** ne pouvaient pas être représentées de manière homogène. Pour y remédier, nous avons conçu une **solution standardisée** : la création d’un **identifiant unique id\_organisation**, obtenu en concaténant :

* **Le code du département**
* **Le libellé du périmètre (CSP pour la PN ou CGD pour la GN, lorsqu'il existe)**
* **Les lettres "PN" pour la Police Nationale ou "GN" pour la Gendarmerie Nationale**

Cette approche a permis de **garantir une correspondance cohérente** entre les forces de l’ordre et les infractions enregistrées, tout en conservant la granularité nécessaire pour les analyses futures. Une fois cette restructuration effectuée, les données ont été préparées dans un format adapté à leur transformation en **nœuds et relations**, assurant ainsi une **migration fluide et efficace** vers Neo4j.

***4.2 Script Python d’intégration des données SQL***

Avant de procéder à la migration vers Neo4j, les données issues du fichier **Excel** contenant les statistiques des crimes et délits en France (2012-2022) ont été extraites et structurées pour être insérées dans une base de données **relationnelle SQLite (infractions.db)**. Cette phase permet d’assurer que les données sont **propres, cohérentes et exploitables** avant leur transformation en un modèle graphe.

Le processus d’intégration commence par l’extraction des données via un script Python utilisant la bibliothèque **Pandas**. La fonction principale create\_df() charge le fichier Excel et filtre les feuilles contenant les données des **services de police et de gendarmerie**. Chaque feuille est ensuite analysée pour identifier **l’année de référence**, l’**organisme responsable** (**Police Nationale ou Gendarmerie Nationale**), et les **informations géographiques** telles que le **code du département** et, pour la Police Nationale, le **périmètre (CSP - Commissariats de Sécurité Publique)**. Comme la **Gendarmerie Nationale ne possède pas de périmètres comparables**, une solution a été mise en place pour uniformiser les données : la création d’un **identifiant unique id\_organisation**, composé du **code département**, du **libellé CGD/CSP** et d’un suffixe indiquant l’organisme ("PN" ou "GN"). Cette standardisation permet d’assurer une **correspondance homogène entre les différentes structures d’intervention**.

Une fois les données extraites, elles sont stockées dans un **DataFrame Pandas** qui servira de base pour l’insertion dans SQLite. Chaque entité principale est ensuite insérée dans sa table respective en supprimant les doublons afin d’éviter des conflits de redondance. Les **départements** sont insérés dans la table départements, les **années d’observation** dans annee, et les **forces de l’ordre** dans force\_odre. Les **infractions** sont également enregistrées, associant un identifiant unique code\_infr à un libellé décrivant le type de crime.

Ensuite, les **brigades** sont insérées dans la table brigade, qui relie les forces de l’ordre aux départements et aux périmètres. Cette table est essentielle car elle permet de structurer la relation entre les crimes enregistrés et l’organisme responsable. Enfin, les **faits criminels** sont insérés dans la table fait, qui contient les informations relatives aux infractions constatées par année, département et brigade, ainsi que le **nombre de faits** enregistrés pour chaque combinaison.

Une attention particulière a été portée à la correction des valeurs manquantes. Certains départements n’ayant pas de nom renseigné dans le fichier source, une valeur par défaut **"Inconnu"** a été attribuée à ces enregistrements afin d’éviter des erreurs lors de la migration. De plus, pour assurer une **intégrité des relations**, l’insertion des faits a été réalisée en utilisant la méthode INSERT OR IGNORE, empêchant ainsi l’ajout de doublons.

Ce traitement a permis de structurer les données sous une forme relationnelle normalisée avant leur transformation vers le modèle graphe. L’ensemble des données étant désormais stocké dans **SQLite (infractions.db)**, la prochaine étape consiste à les convertir en **nœuds et relations** dans Neo4j afin d’optimiser leur exploitation et leur analyse.

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

***4.3 Migration des données vers Neo4j***

Une fois les données nettoyées et préparées, elles ont été insérées dans Neo4j sous forme de nœuds interconnectés.

**Le modèle retenu repose sur les entités suivantes :**

*Infraction* : Identifie les différents types de crimes et délits.

*Département* : Définit les zones géographiques où les faits criminels sont enregistrés.

*ForceOrdre* : Représente l'autorité responsable (Police Nationale ou Gendarmerie Nationale).

*Annee* : Permet d’étudier l’évolution des crimes sur la période 2012-2022.

*Brigade* : Précise l’unité opérationnelle ayant enregistré une infraction.

*Fait* : Représente une infraction constatée, avec son année et son volume.

**Ces nœuds sont reliés entre eux par les relations suivantes** :

(Brigade)-[:APPARTIENT\_A]->(Département) : Associe une brigade à son département de rattachement.

(Brigade)-[:FAIT\_PARTIE\_DE]->(ForceOrdre) : Relie une brigade à la Police Nationale ou Gendarmerie Nationale.

(Brigade)-[:A\_ENREGISTRÉ]->(Fait) : Indique qu'une brigade a enregistré une infraction.

(Fait)-[:CONCERNE]->(Infraction) : Permet d’identifier le type d’infraction concerné.

(Fait)-[:A\_LIEU\_DANS]->(Département) : Associe une infraction à son département d’occurrence.

(Infraction)-[:CONSTATE\_PAR]->(ForceOrdre) : Définit quelle autorité a enregistré l’infraction.

(Fait)-[:SUR\_ANNEE]->(Annee) : Permet d’étudier l’évolution des infractions dans le temps.

Ce modèle offre une meilleure visualisation et analyse des relations criminelles, facilitant des recherches rapides sur les connexions entre crimes, forces de l’ordre et zones géographiques.

**Validation et Comparaison des Résultats**

***5.1 Exemples de Requêtes Cypher et Résultats***

Nombre total de crimes par département :

*MATCH (f:Fait)-[:A\_LIEU\_DANS]->(d:Departement)*

*RETURN d.nom\_dep, SUM(f.nb\_infraction) AS total\_crimes*

*ORDER BY total\_crimes DESC*

*Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Le contenu généré par l’IA peut être incorrect.*

Types de crimes les plus fréquents par département :

*MATCH (f:Fait)-[:CONCERNE]->(i:Infraction), (f)-[:A\_LIEU\_DANS]->(d:Departement) RETURN d.code\_dep, i.lib\_infr, SUM(f.nb\_infraction) AS nb\_faits ORDER BY d.code\_dep, nb\_faits DESC*

*Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Le contenu généré par l’IA peut être incorrect.*

***5.2 Analyse des Performances***

L’un des principaux objectifs de cette migration était d’optimiser l’exécution des requêtes analytiques en réduisant la complexité des jointures SQL.

Dans une base relationnelle (SQLite), une requête typique pour récupérer le nombre total de crimes par département nécessitait plusieurs jointures entre les tables fait, brigade, départements et infractions :

***SQL :***

*SELECT d.code\_dep, i.lib\_infr, SUM(f.nb\_infraction) AS nb\_faits*

*FROM fait f*

*JOIN brigade b ON f.id\_brigade = b.id\_brigade*

*JOIN départements d ON b.code\_dep = d.code\_dep*

*JOIN infractions i ON f.code\_infr = i.code\_infr*

*GROUP BY d.code\_dep, i.lib\_infr*

*ORDER BY d.code\_dep, nb\_faits DESC;*

*Une image contenant texte, capture d’écran, nombre, affichage

Le contenu généré par l’IA peut être incorrect.*

***NEO4J :***

*MATCH (f:Fait)-[:CONCERNE]->(i:Infraction), (f)-[:A\_LIEU\_DANS]->(d:Departement)*

*RETURN d.code\_dep, i.lib\_infr, SUM(f.nb\_infraction) AS nb\_faits*

*ORDER BY d.code\_dep, nb\_faits DESC*

*Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.*

On prouve encore une fois que le choix de modèle en graphe est beaucoup mieux car les requêtes sont moins longues, moins couteuses et plus adaptées.

**Conclusion**

Le modèle graphe, basé sur Neo4j, se révèle particulièrement adapté pour des analyses relationnelles complexes. En permettant une représentation claire et détaillée des liens entre les différentes entités, il offre une grande souplesse pour l’ajout et l’évolution des relations au fil du temps. Cette flexibilité est un atout majeur, notamment pour adapter le modèle aux besoins futurs d’analyse.

En comparaison avec le modèle relationnel initial, la base orientée graphe permet une navigation fluide entre les entités, réduisant considérablement le coût des jointures complexes observé en SQL. Grâce à Neo4j, il devient possible d’explorer efficacement **les relations indirectes** entre infractions, forces de l’ordre et zones géographiques, offrant ainsi un **cadre d’analyse plus riche et plus performant**. L’optimisation des requêtes permet également d’obtenir des **résultats plus rapides**, notamment pour les analyses impliquant des connexions multiples entre les entités, comme l’évolution des infractions par département ou la coopération entre brigades.

L’**uniformisation des données** entre la **Police Nationale (PN)** et la **Gendarmerie Nationale (GN)**, grâce à la création d’un **identifiant unique id\_organisation**, a permis de lever les incohérences initiales du modèle relationnel. Cette normalisation garantit une exploitation homogène des données, quelle que soit l’organisation qui enregistre les infractions. De plus, la **réduction du volume de stockage des relations** et la **modularité du graphe** rendent le système plus évolutif et mieux adapté aux besoins futurs d’analyse, notamment pour intégrer de **nouvelles sources de données** ou affiner l’étude des **réseaux criminels**.

Enfin, cette migration ouvre la voie à des **perspectives d'amélioration et d'extension**. Par exemple, des analyses plus poussées pourraient être réalisées en intégrant **des données socio-économiques**, **des cartes interactives**, ou encore **des modèles prédictifs** pour identifier les tendances criminelles futures. L’approche graphe permettrait aussi de **simuler des scénarios**, comme l’impact d’une augmentation des effectifs policiers sur la répartition des infractions.

En conclusion, la transformation de notre base relationnelle en un modèle orienté graphe avec Neo4j s'est révélée être une **solution efficace et évolutive**, permettant d’optimiser l’analyse des données criminelles en France tout en offrant une architecture flexible et performante pour les besoins futurs.