به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس آزمایشگاه پایگاه‌داده

پیش‌گزارش سوم

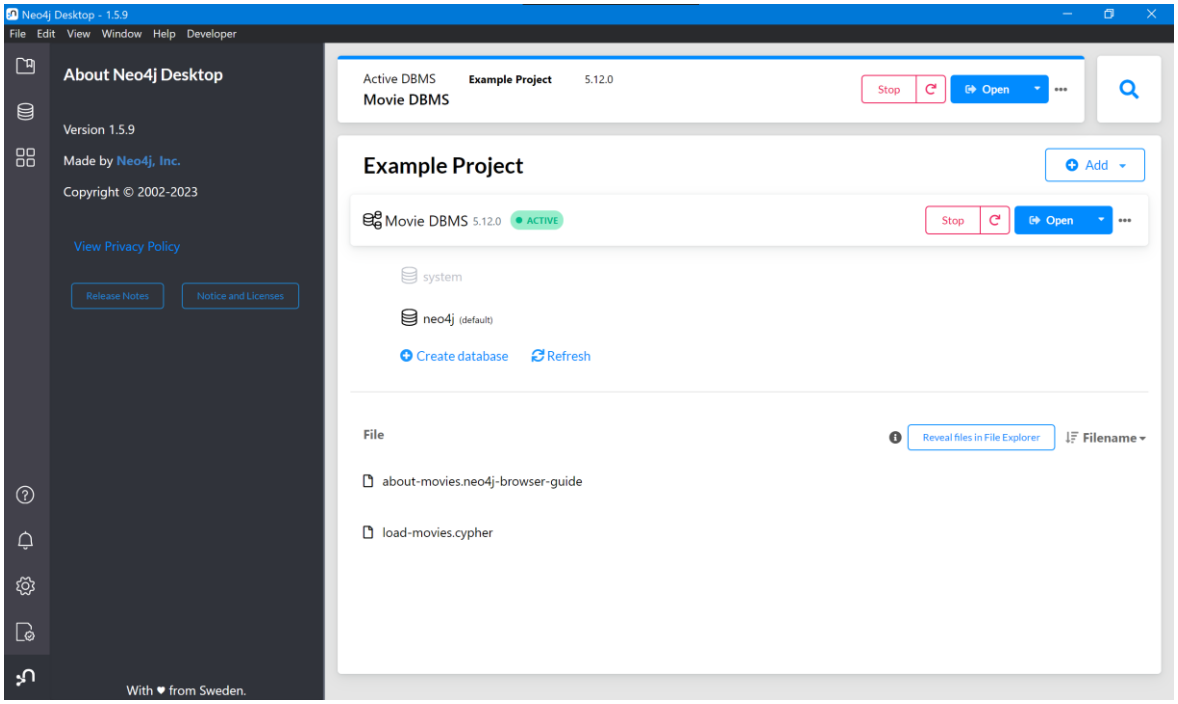
نام و نام خانوادگی	میثاق محقق
شماره دانشجویی	**810199484**
تاریخ ارسال گزارش	**1402.08.07**

فهرست

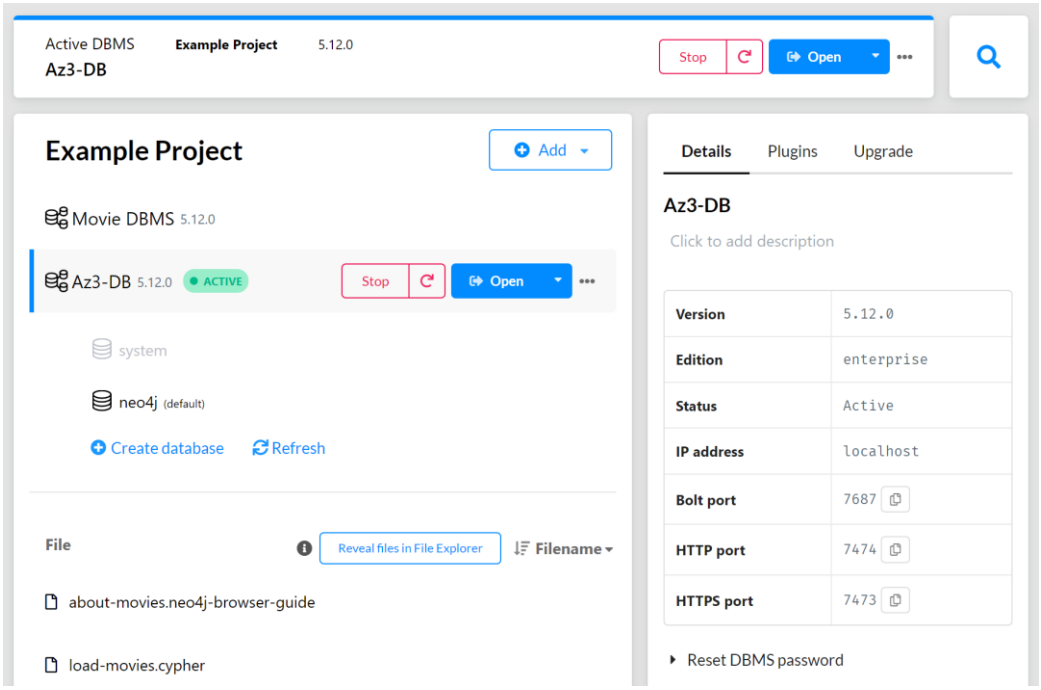
پاسخ 1. آشنایی با Neo4j

1-1. نصب Neo4j

برنامه نصب شد:

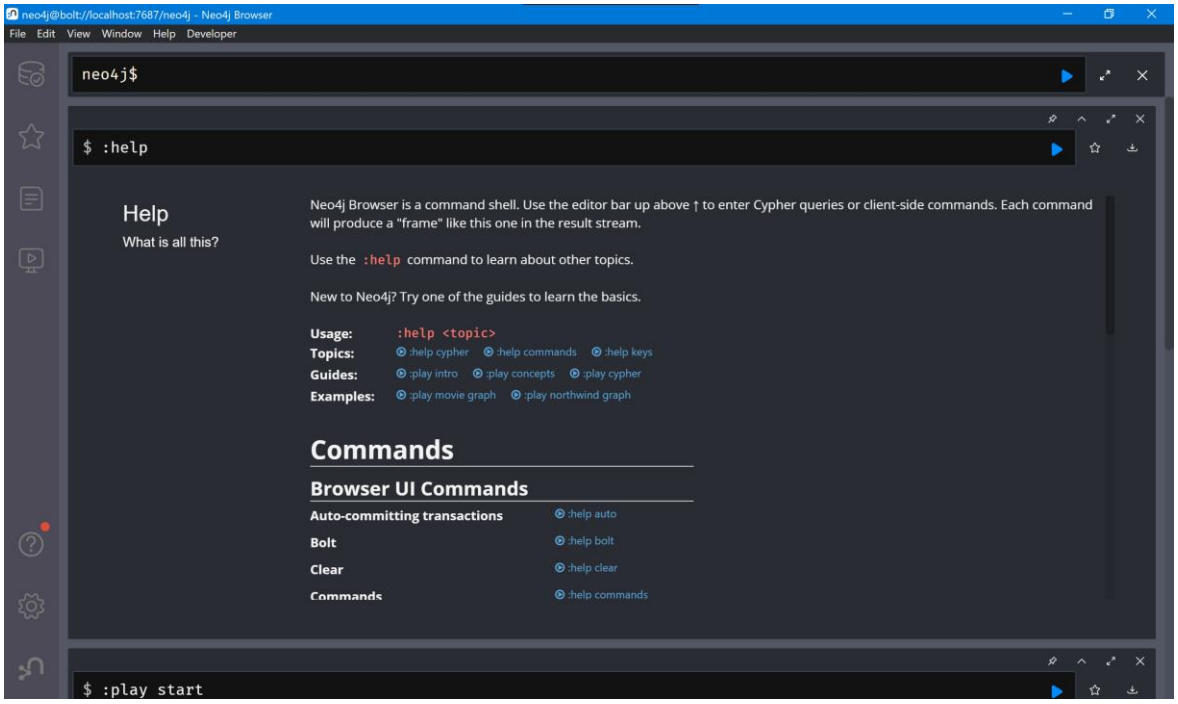


یک دیتابیس جدید به نام Az3-DB ساخته شد:



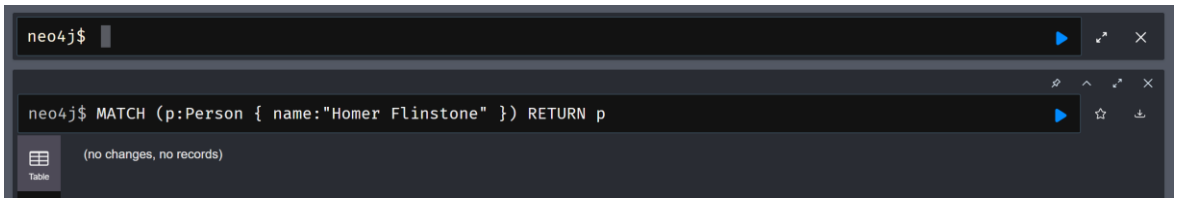
1-2. مرورگر Neo4j

در اینجا می‌شود کامند و Cypher-ها را وارد کرده و روی دیتابیس اعمال کرد. سایفرها شبیه کوئری‌ها در RDBMS-ها اند. مثلا کامند help: خروجی زیر را دارد. و clear: خروجی‌ها را پاک می‌کند.



1-3. آشنایی با Cypher

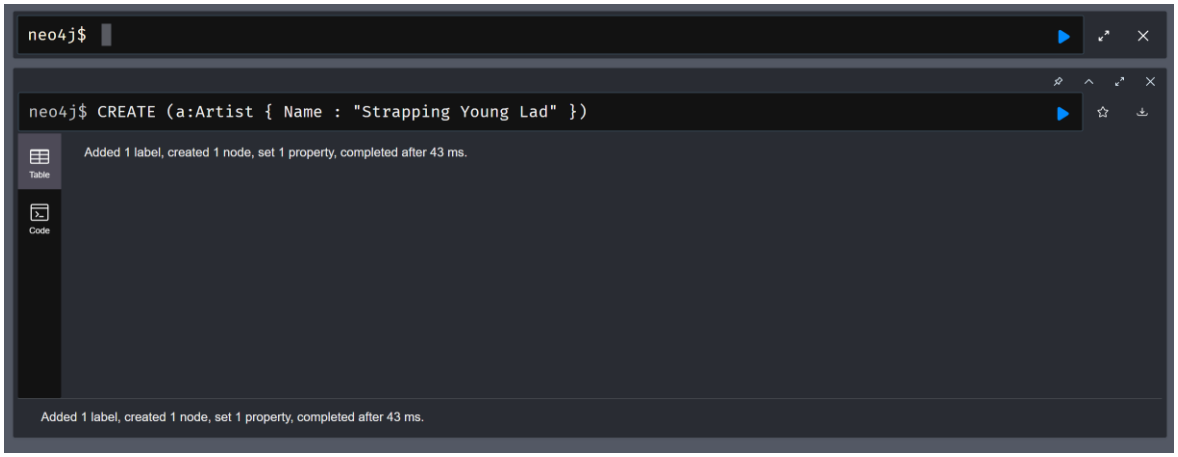
زبان کوئری Cypher مشابه با SQL است.



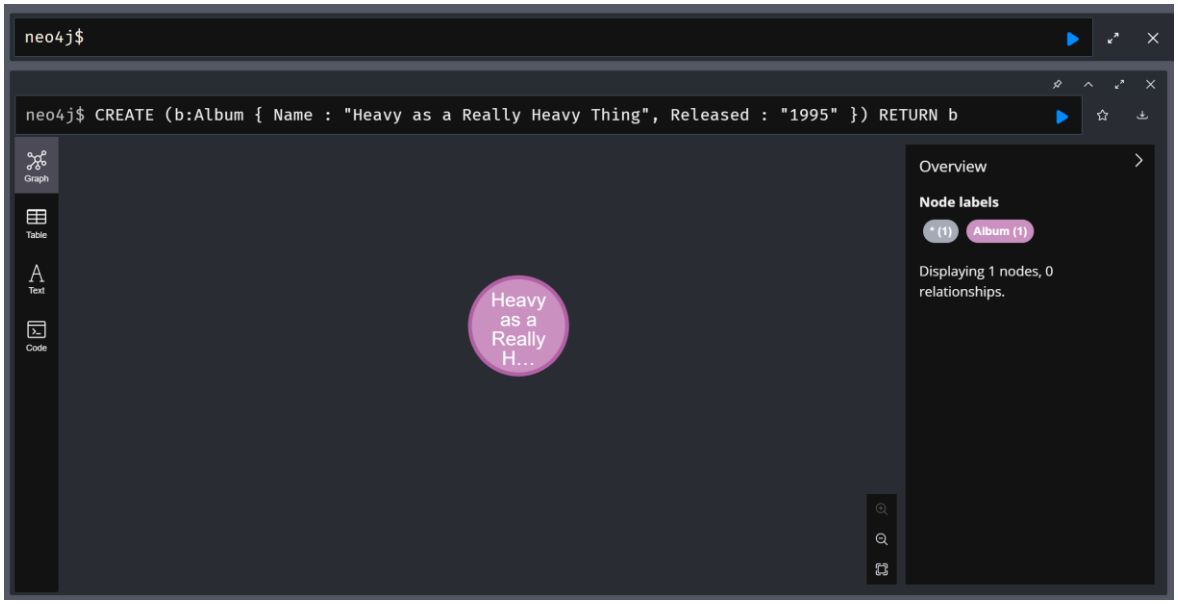
فردی به نام Homer Flinstone وجود ندارد.

1-4. ساخت Node

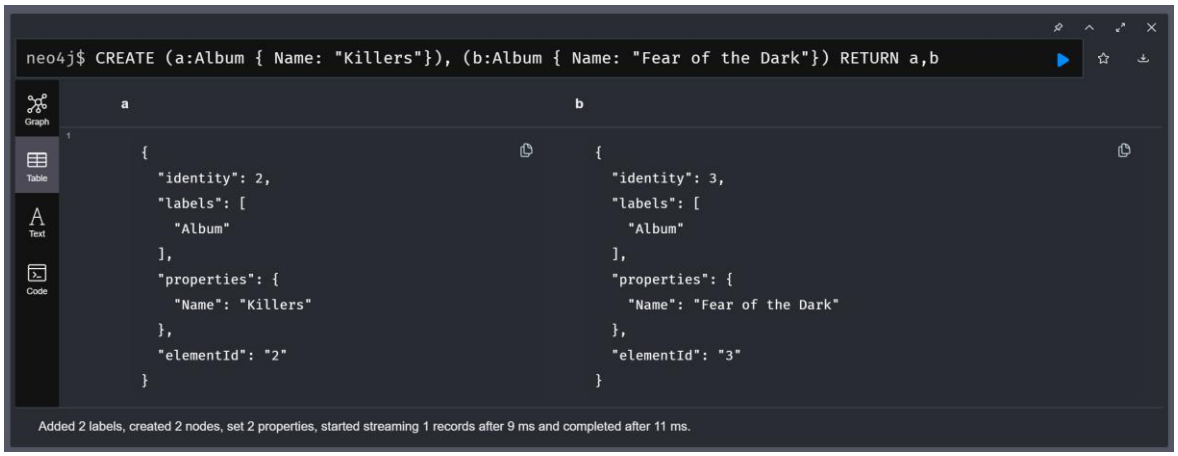
یک Artist و یک Album می‌سازیم:



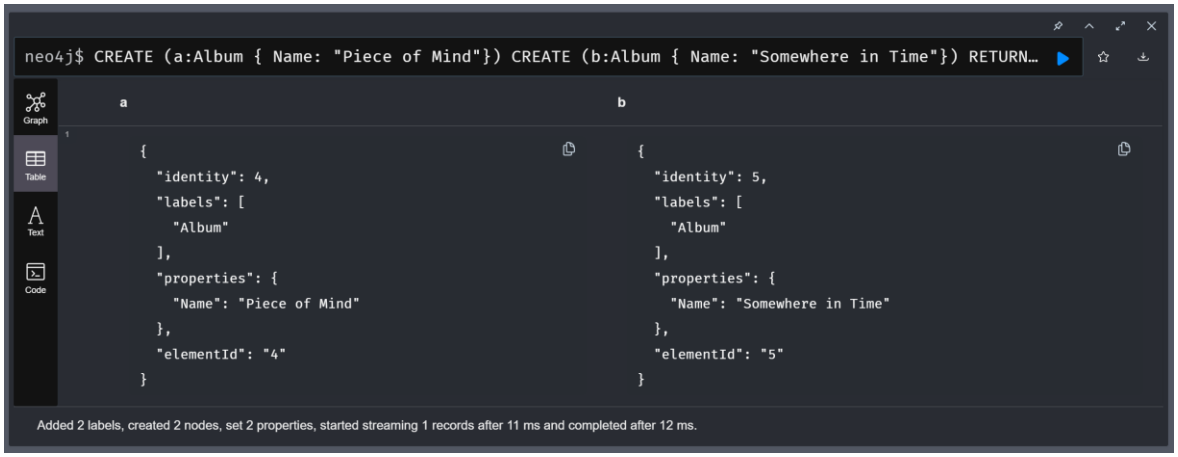
در این مثال، a نام node شده که label آن Artist است.



می‌توان چند node باهم ساخت (با کاما) و نوع نمایش جدولی را دید:

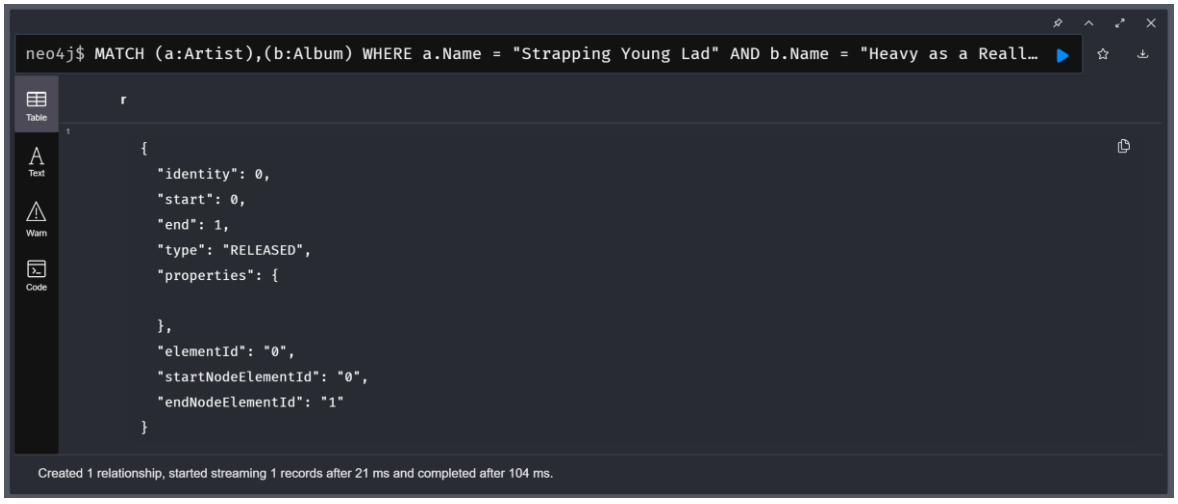


می‌توان به جای کاما از چند create هم استفاده کرد:

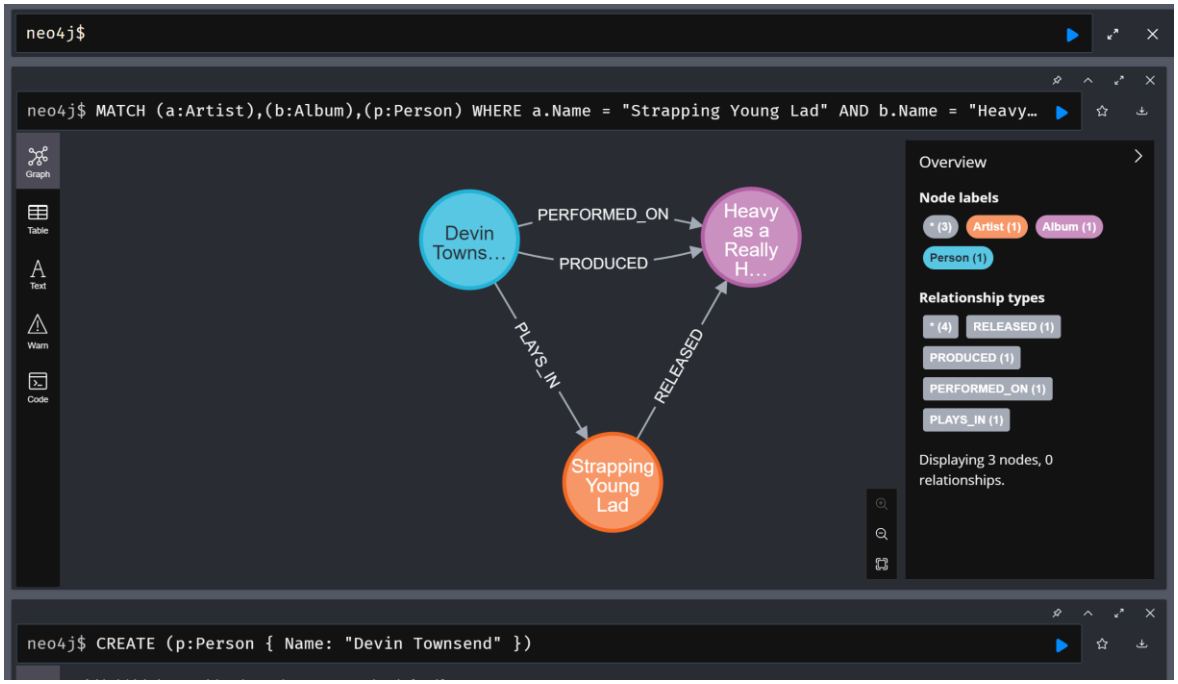


5-1. ساخت رابطه

با استفاده از `(a)-[r:RELEASED]->(b)` می‌توان یک رابطه از a به b ساخت. داخل پرانتز به یک node اشاره می‌کند.



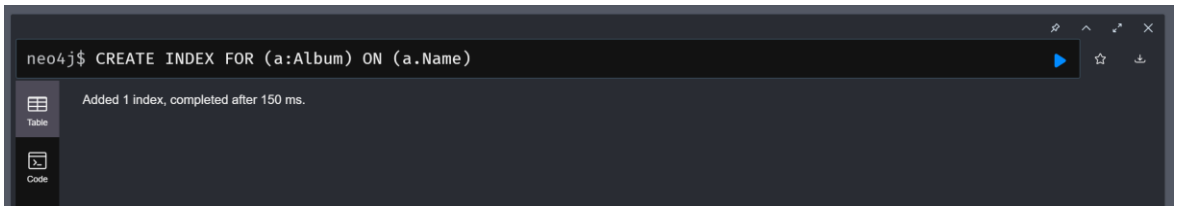
حال Person node ای به نام Devin Townsend ساخته و سه رابطه تعریف می‌کنیم:



رابطه released در کامند قبل ساخته شده بود.

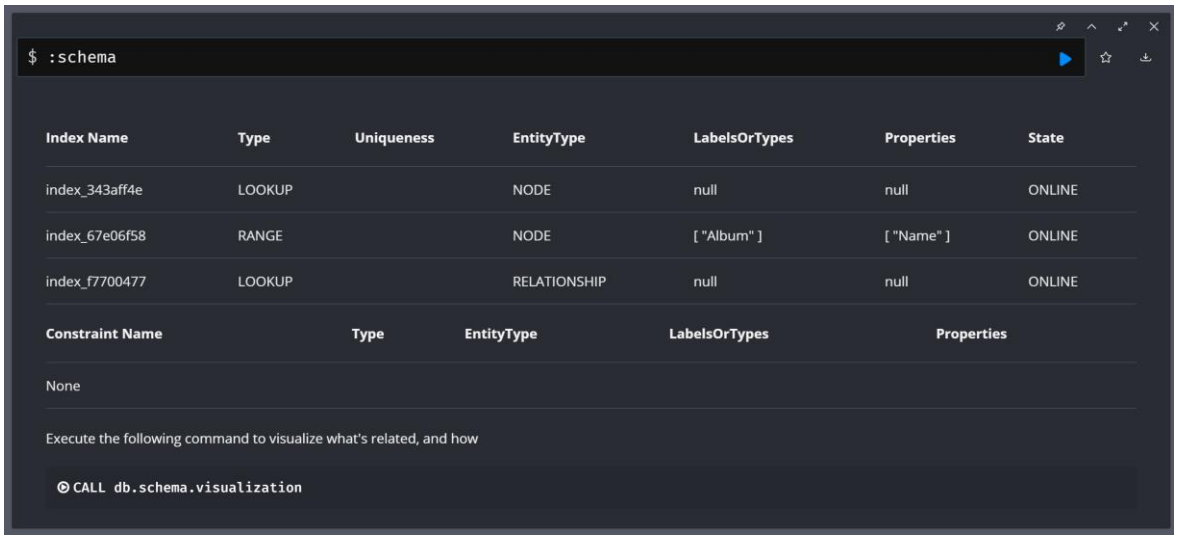
6-1. ساخت Index

روی name آلبوم ایندکس جهت بازیابی سریع‌تر می‌سازیم:



سایفر انجام این کار با لینک داده شده متفاوت بوده و با سرچ کردن سینتکس جدید آن نوشته شده است.

با schema: می‌توان همه ایندکس‌ها را دید:



می‌توان دستی از index-ای خاص استفاده کرد:

7-1. ساخت Constraint

دو نوع constraint می‌شود ساخت. uniqueness constraint و property existence constraint

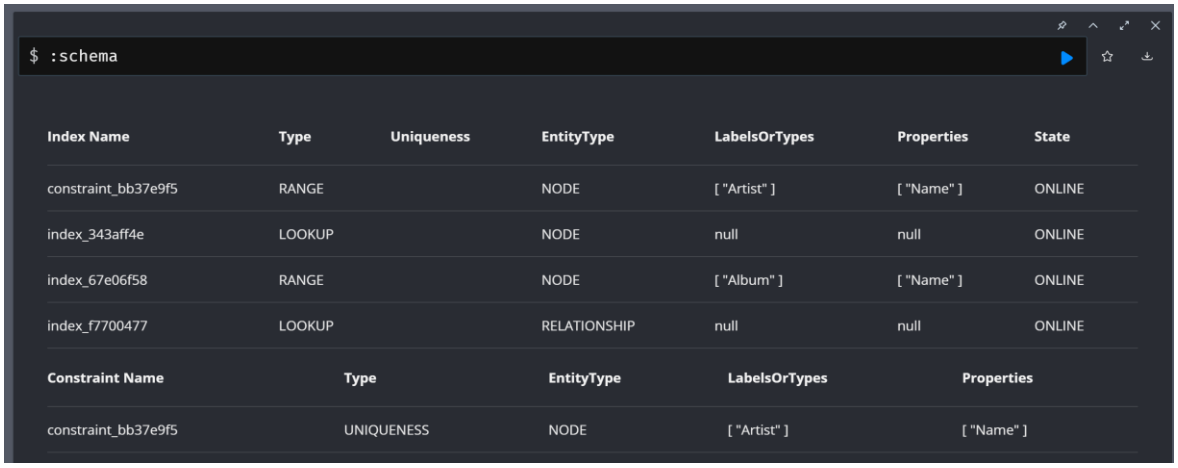
اولی یعنی در بین node-های یک label، هیچ دو نباید یک property با مقدار یکسان داشته باشند.

دومی یعنی در بین node-های یک label، همه شان باید یک property خاص را داشته باشند.

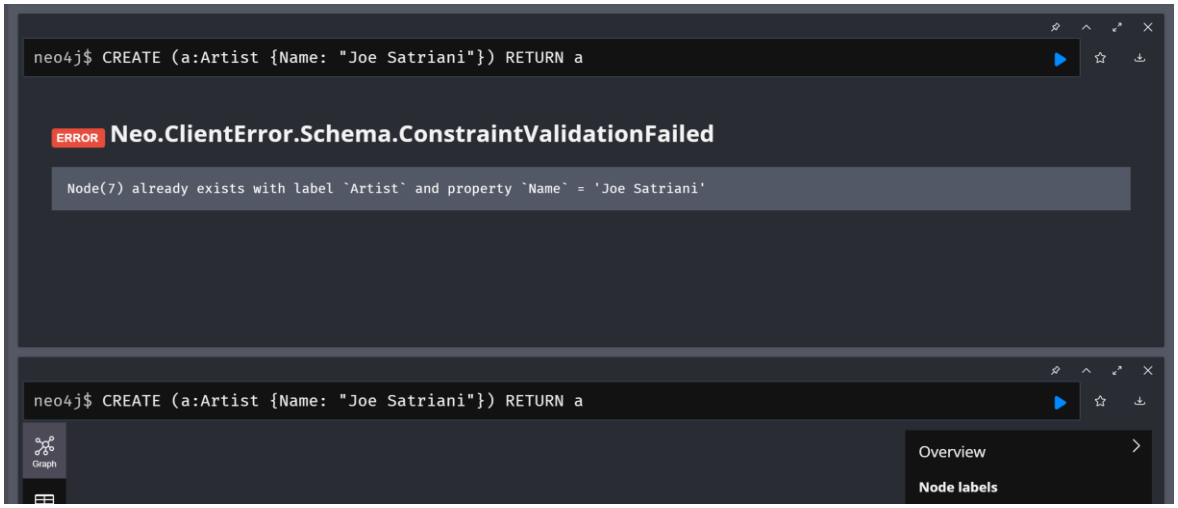
اینجا نیز مانند index کم سینتکس از سایت داده شده متفاوت می‌باشد.

در اینجا یک uniqueness constraint روی اسم آرتیست ساخته شده است.

با schema: می‌توان مانند index-ها، constraint-ها هم دید.

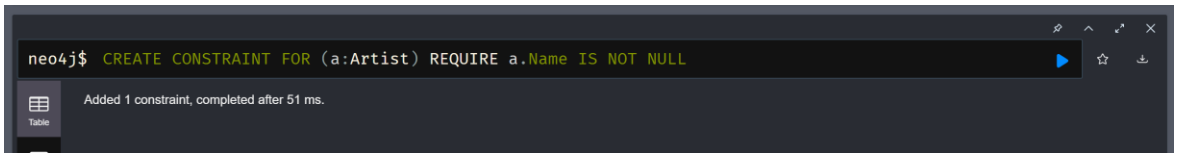


حال آن را با اضافه کردن آرتیست با نام تکراری تست می‌کنیم:

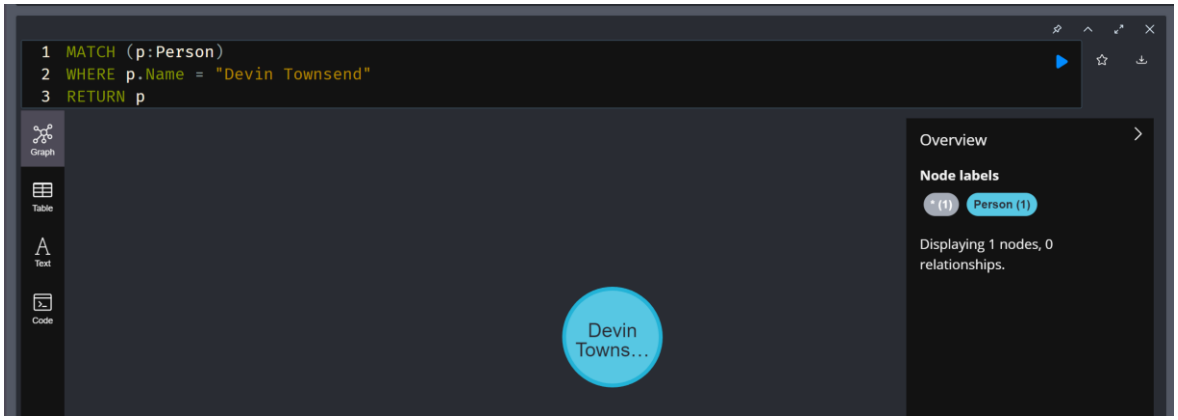


بار اول اجرا می‌شود و بار دوم ConstraintValidationFailed می‌دهد.

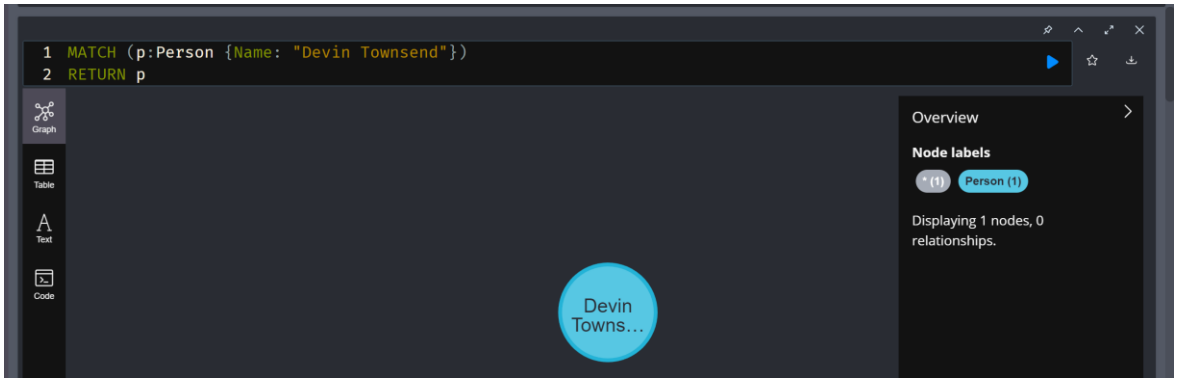
حال یک property existence constraint می‌سازیم.



8-1. انتخاب با MATCH



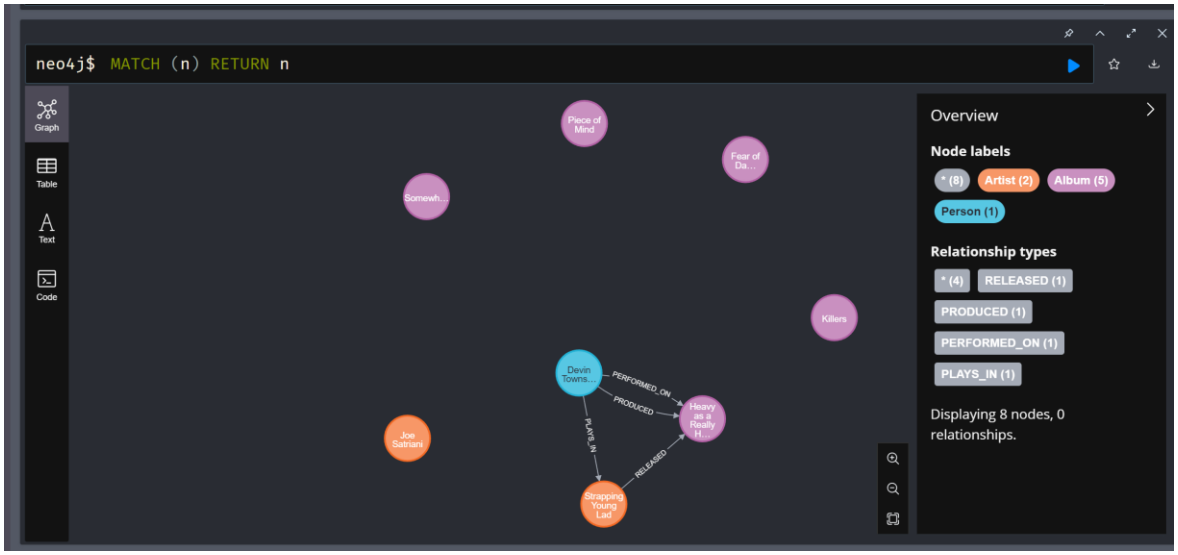
در اینجا از where استفاده شد که مشابه sql است. می‌توان بدون آن هم نوشت:



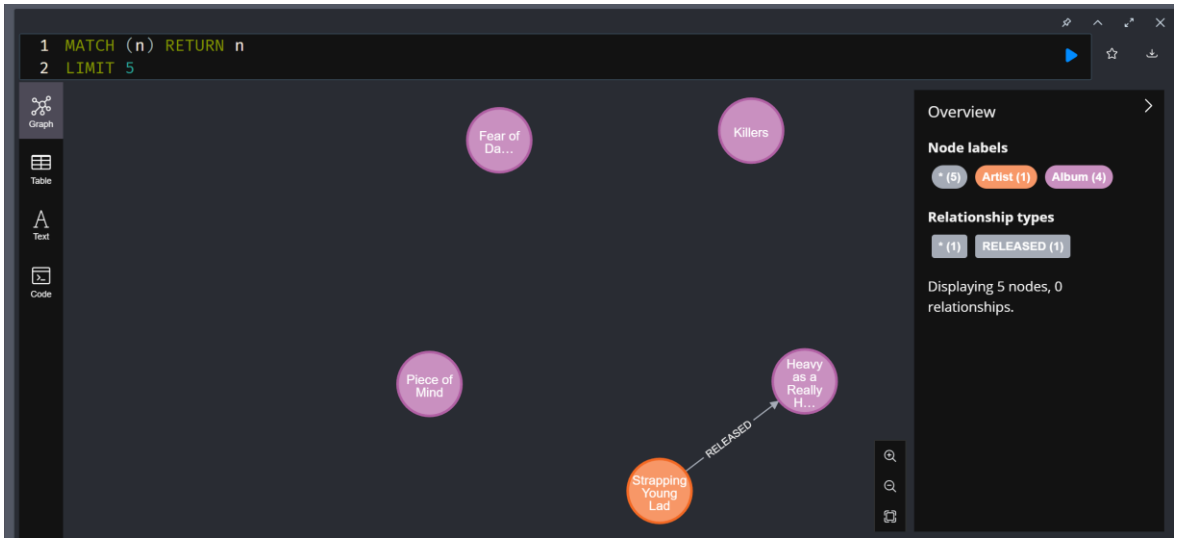
می‌توان علاوه بر node-ها، relationship-ها هم MATCH کرد.

از شکل cypher می‌توان مفهوم کار انجام شده را به راحتی دید.

با دستور زیر می‌توان همه node-ها را دید:

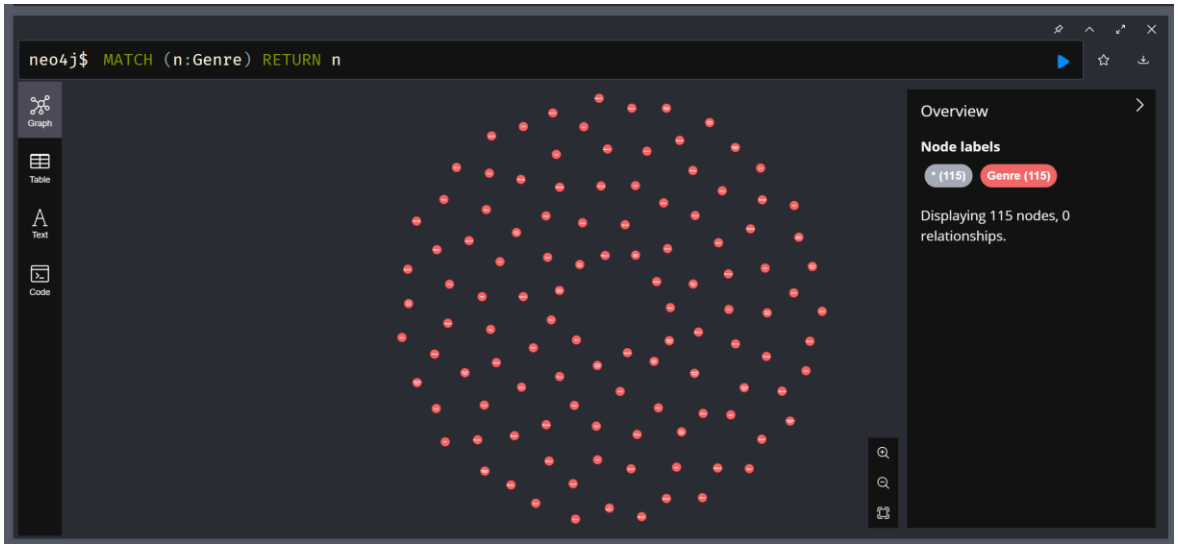


می‌شود خروجی را لیمیت کرد:

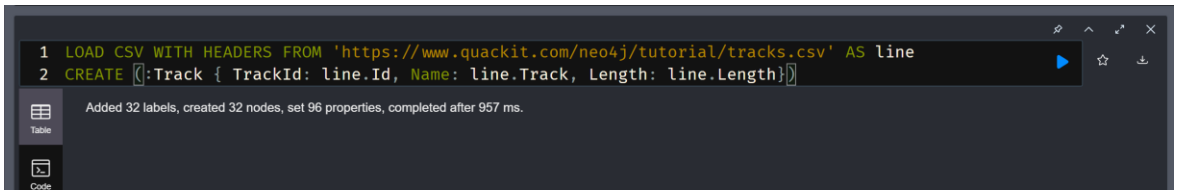


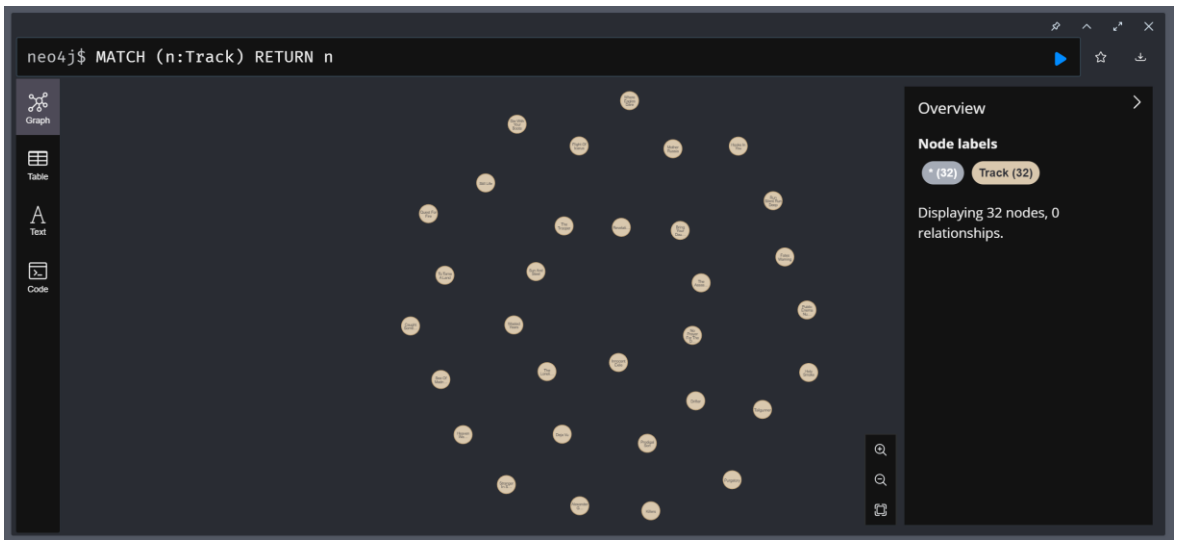
9-1. ایمپورت داده از CSV

برای لود از فایل لوکال باید از file:/// استفاده کرد.

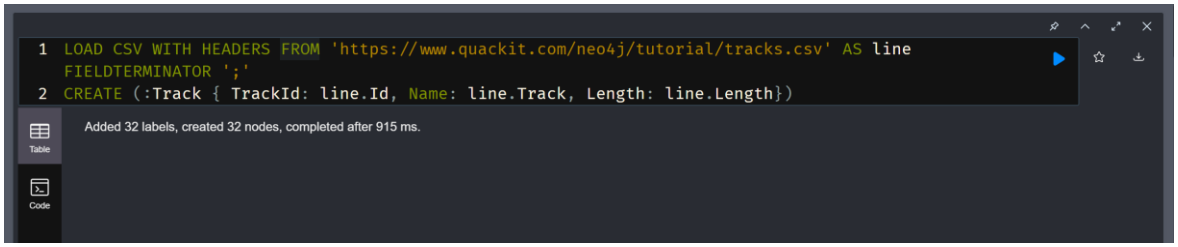


اگر csv هدر داشته باشد (قبلی نداشت) می‌توان به صورت زیر انجام داد:

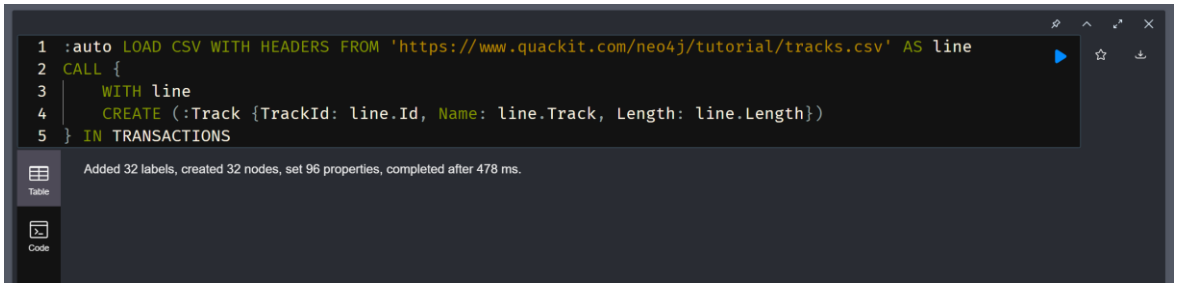




می‌توان به جای کاما از کاراکتر دیگری استفاده کرد در csv:



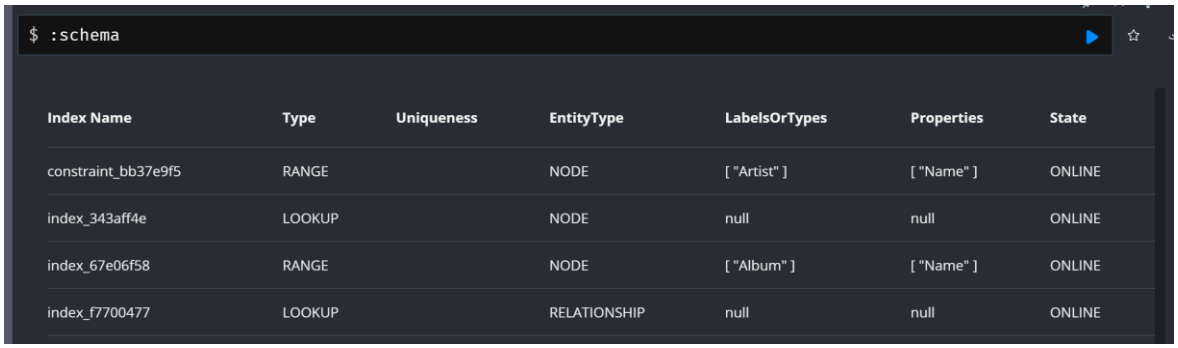
برای داده‌های بزرگ از روش زیر استفاده می‌کنیم که سینتکس آن متفاوت از سایت است:



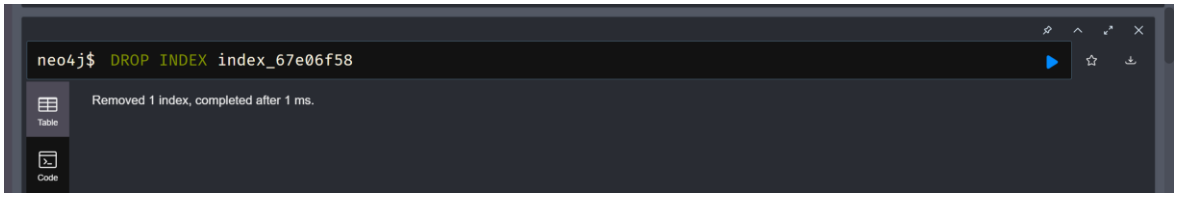
برای تغییر periodic commit به 800 در آخر کامند بالا OF 800 ROWS اضافه می‌کنیم.

10-1. حذف Index

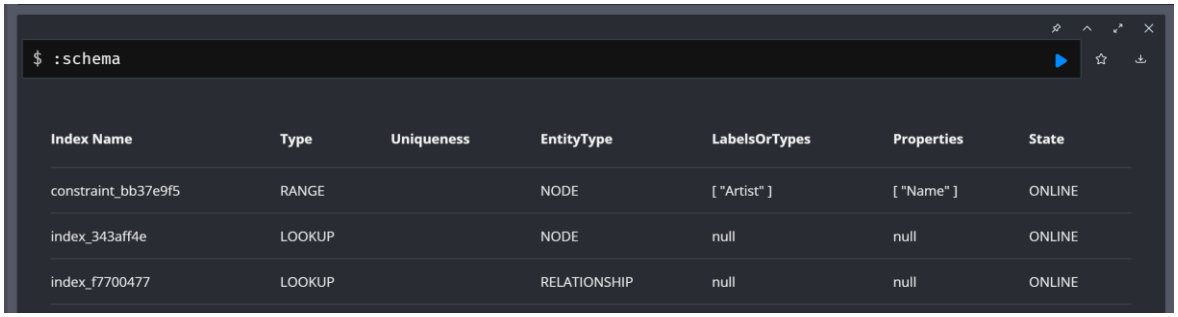
سینتکس سایت کار نکرده و با اسم اصلی ایندکس آن را حذف می‌کنیم:



دنبال ایندکس روی نام آلبوم هستیم پس index_67e06f58 را می‌خواهیم:



و می‌بینیم که حذف شده است:



11-1. حذف Constraint

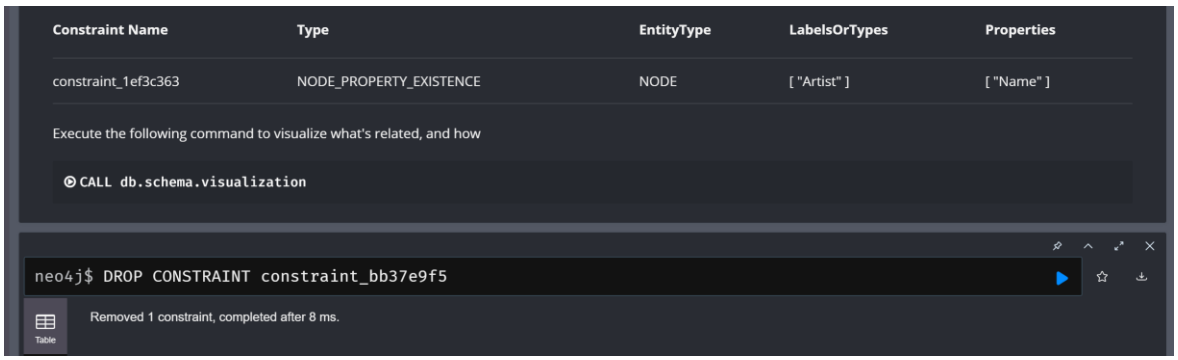
مانند حذف Index، از نام constraint استفاده می‌کنیم.

Constraint Name	Type	EntityType	LabelsOrTypes	Properties
constraint_1ef3c363	NODE_PROPERTY_EXISTENCE	NODE	["Artist"]	["Name"]
constraint_bb37e9f5	UNIQUENESS	NODE	["Artist"]	["Name"]

از آنجا که کامند داده شده

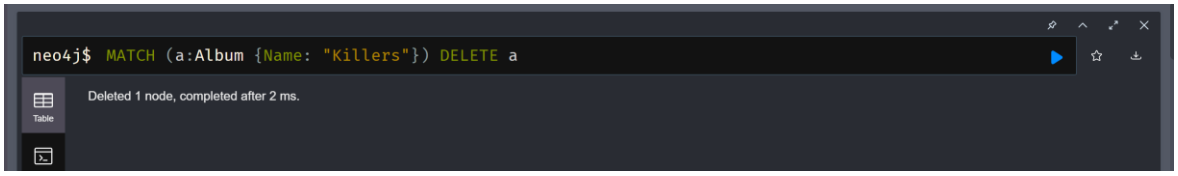
DROP CONSTRAINT ON (a:Artist) ASSERT a.Name IS UNIQUE

باید uniqueness را حذف کنیم که constraint_bb37e9f5 است.

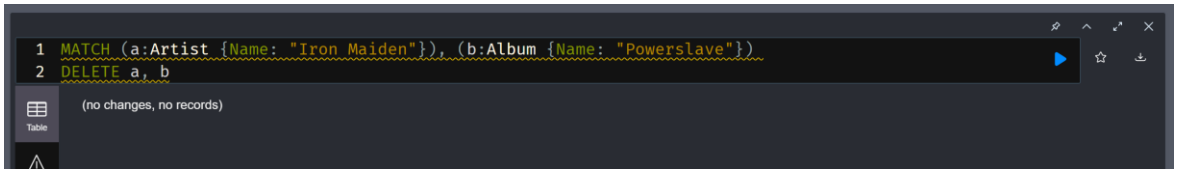


12-1. حذف Node

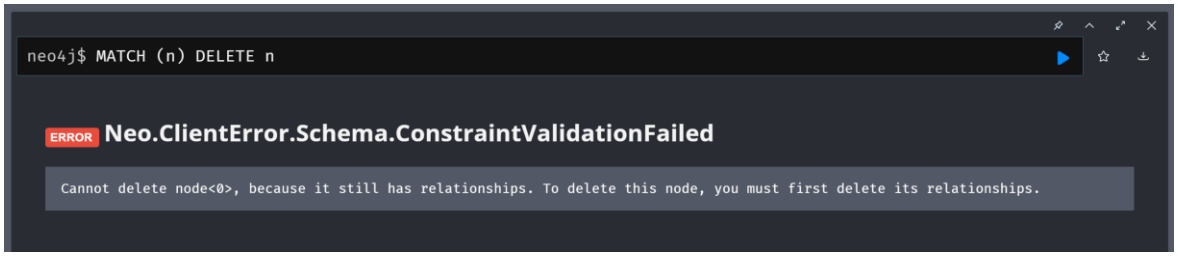
آلبوم‌ها با نام Killers را حذف می‌کنیم.



حذف چند node با هم:



حذف همه node-ها:



از آنجا که رابطه‌ها باید ابتدا حذف شوند، این عمل ممکن نیست.

12-1. حذف رابطه

```
neo4j$ MATCH ()-[r:RELEASED]-() DELETE r
```

Deleted 1 relationship, completed after 28 ms.

```
1 MATCH (:Artist)-[r:RELEASED]-(:Album)
2 DELETE r
```

(no changes, no records)

```
1 MATCH (:Artist {Name: "Strapping Young Lad"})-[r:RELEASED]-(:Album {Name: "Heavy as a Really
  Heavy Thing"})
2 DELETE r
```

(no changes, no records)

اینجا طی سه مرحله در حذف رابطه specific تر شدیم.

همانطور که دیدیم node با رابطه را نمی‌شود حذف کرد:

```
neo4j$ MATCH (a:Artist {Name: "Strapping Young Lad"}) DELETE a
```

ERROR **Neo.ClientError.Schema.ConstraintValidationFailed**

Cannot delete node<0>, because it still has relationships. To delete this node, you must first delete its relationships.

با استفاده از detach delete می‌توانیم بدون حذف دستی رابطه‌های آن، آن را حذف کنیم.

```
neo4j$ MATCH (a:Artist {Name: "Strapping Young Lad"}) DETACH DELETE a
```

Deleted 1 node, deleted 1 relationship, completed after 2 ms.

حال می‌توان کل دیتابیس را پاک کرد:

```
neo4j$ MATCH (n) DETACH DELETE n
```

Deleted 217 nodes, deleted 2 relationships, completed after 3 ms.