



به نام خدا
دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر



درس آزمایشگاه پایگاه داده پیش گزارش سوم

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

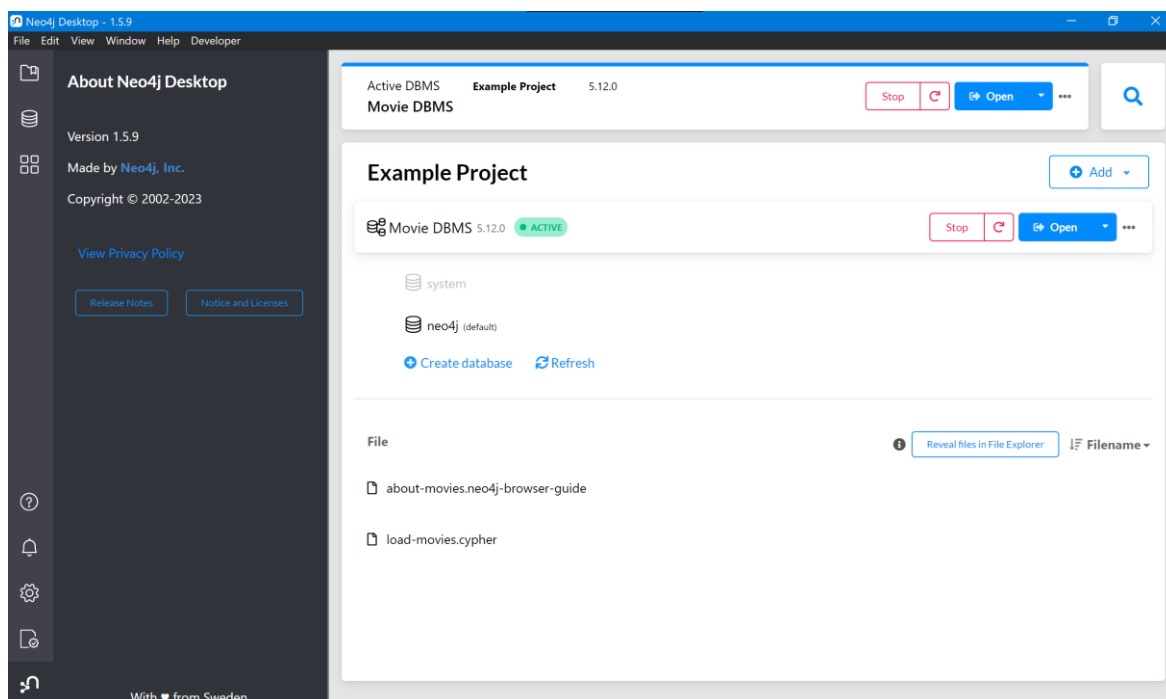
فهرست

2	پاسخ 1. آشنایی با Neo4j
2-1	1-1. نصب Neo4j
3-1	2-1. مرورگر Neo4j
3	3-1. آشنایی با Cypher
3	4-1. ساخت Node
5	5-1. ساخت رابطه
6	6-1. ساخت Index
7	7-1. ساخت Constraint
9	8-1. انتخاب با MATCH
10	9-1. ایمپورت داده از CSV
12	10-1. حذف Index
12	11-1. حذف Constraint
14	12-1. حذف رابطه

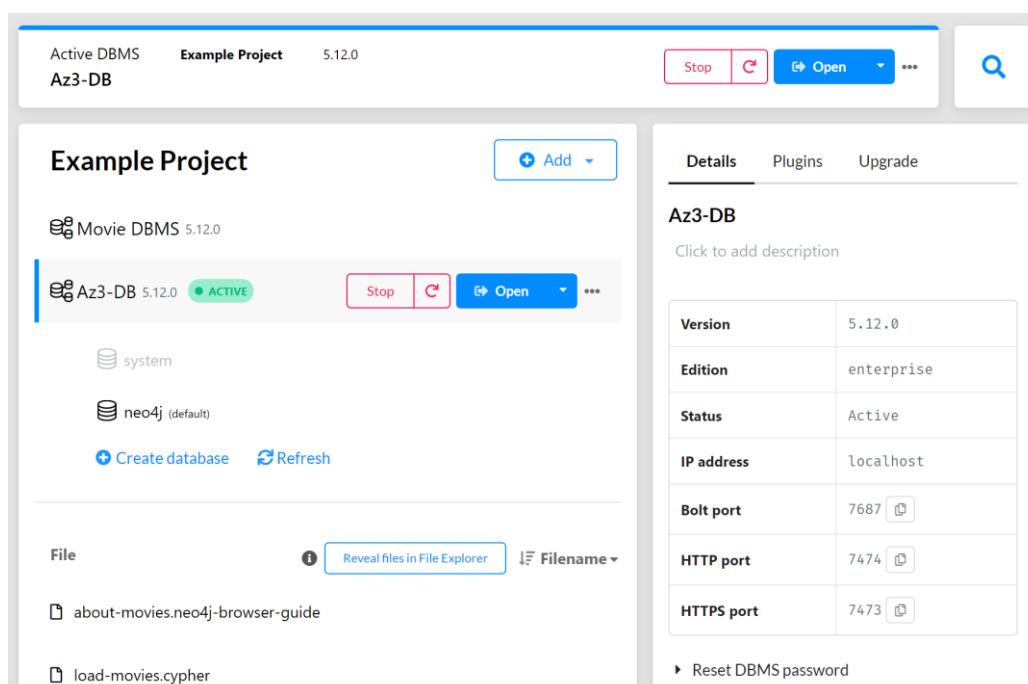
پاسخ 1. آشنایی با Neo4j

1-1. نصب Neo4j

برنامه نصب شد:

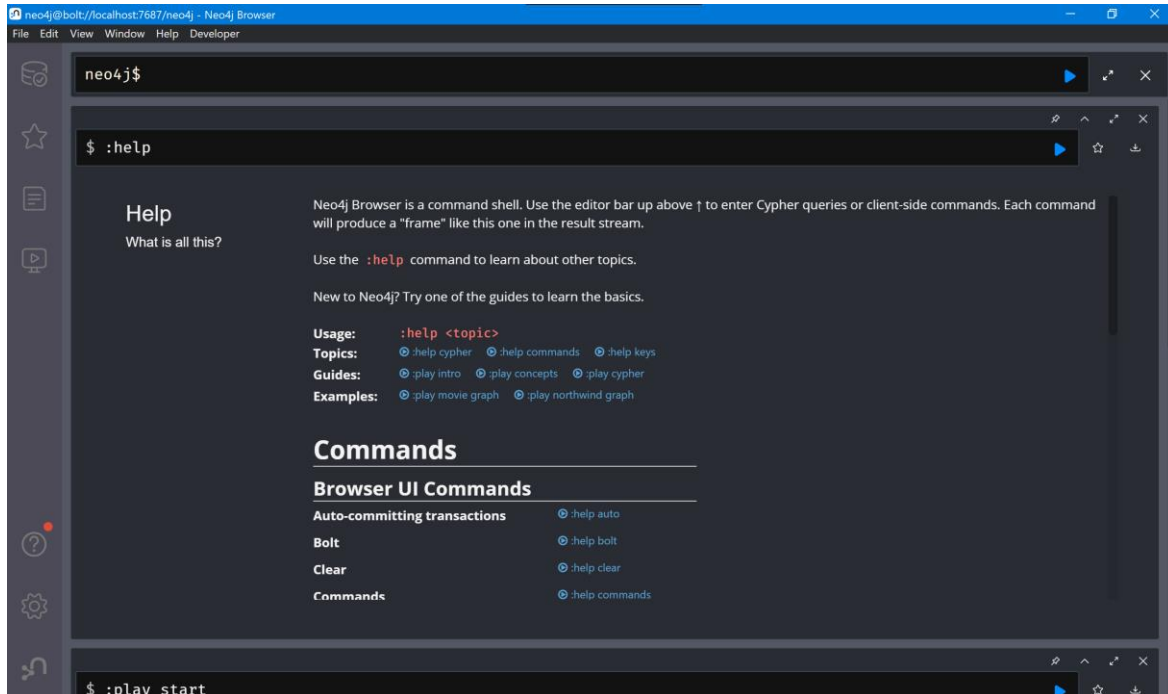


یک دیتابیس جدید به نام Az3-DB ساخته شد:



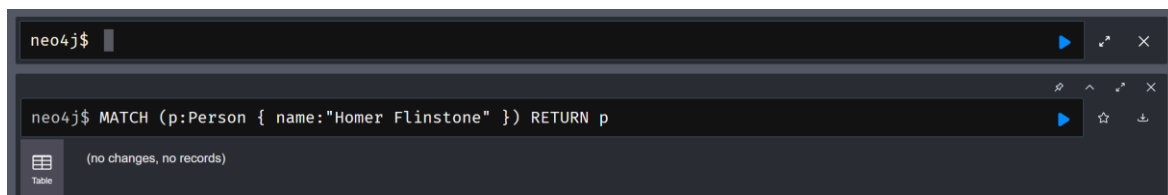
2-1. مرورگر Neo4j

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



3-1. آشنایی با Cypher

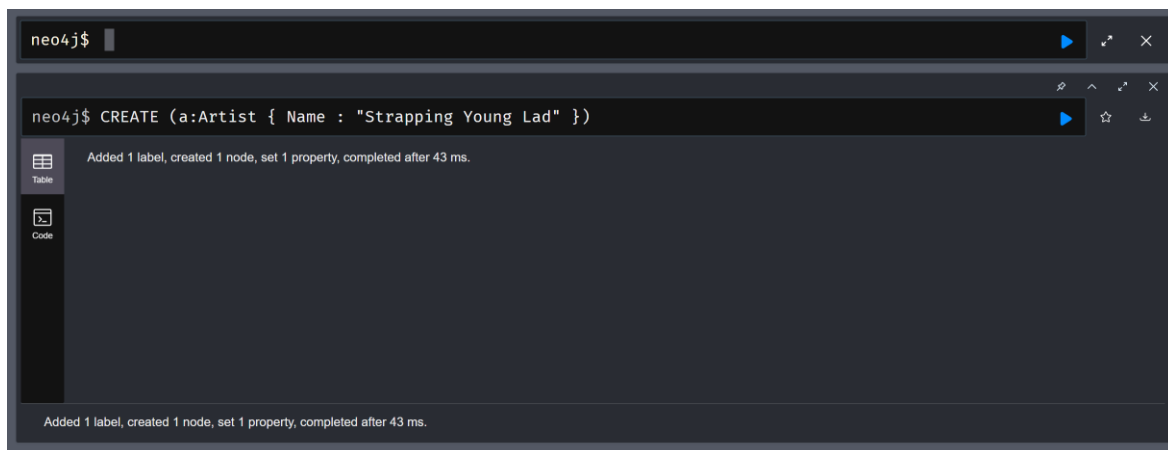
زبان کوئری Cypher مشابه با SQL است.



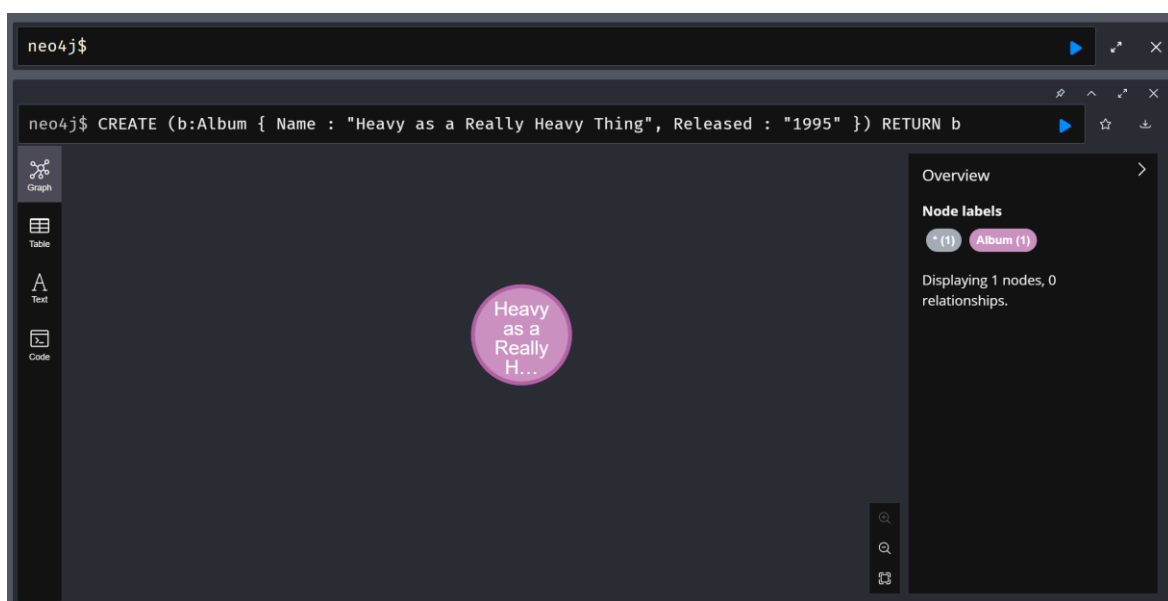
فردی به نام Homer Flinstone وجود ندارد.

4-1. ساخت Node

یک Artist و یک Album می‌سازیم:



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



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



می توان به جای کاما از چند create هم استفاده کرد:

```
neo4j$ CREATE (a:Album { Name: "Piece of Mind"}) CREATE (b:Album { Name: "Somewhere in Time"}) RETURN...
```

a	b
<pre>{ "identity": 4, "labels": ["Album"], "properties": { "Name": "Piece of Mind" }, "elementId": "4" }</pre>	<pre>{ "identity": 5, "labels": ["Album"], "properties": { "Name": "Somewhere in Time" }, "elementId": "5" }</pre>

Added 2 labels, created 2 nodes, set 2 properties, started streaming 1 records after 11 ms and completed after 12 ms.

5-1. ساخت رابطه

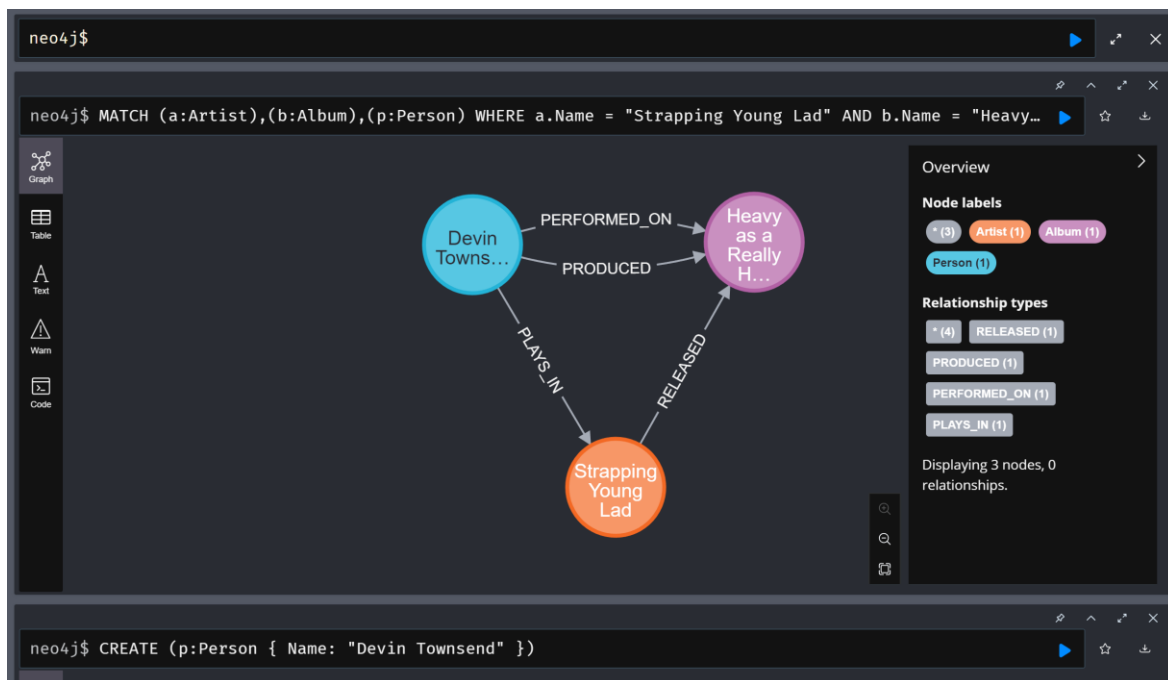
با استفاده از $(a) - [r:RELEASED] -> (b)$ می‌توان یک رابطه از a به b ساخت. داخل پرانتز به یک node اشاره می‌کند.

```
neo4j$ MATCH (a:Artist),(b:Album) WHERE a.Name = "Strapping Young Lad" AND b.Name = "Heavy as a Reall...
```

r
<pre>{ "identity": 0, "start": 0, "end": 1, "type": "RELEASED", "properties": { }, "elementId": "0", "startNodeElementId": "0", "endNodeElementId": "1" }</pre>

Created 1 relationship, started streaming 1 records after 21 ms and completed after 104 ms.

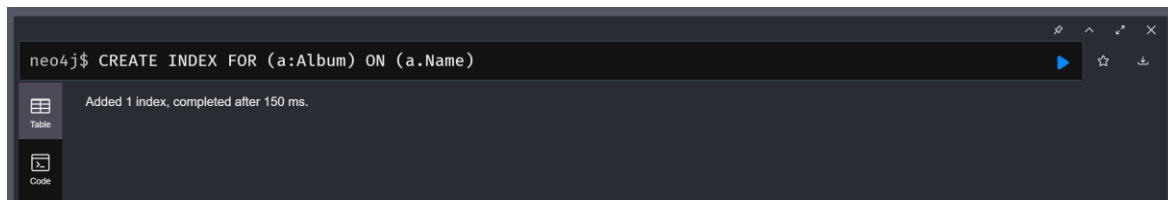
حال Person node ای به نام Devin Townsend ساخته و سه رابطه تعریف می‌کنیم:



رابطه released در کامند قبل ساخته شده بود.

6-1. ساخت Index

روی name آلبوم ایندکس جهت بازیابی سریع تر می سازیم:



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

با schema: می توان همه ایندکس ها را دید:

\$:schema

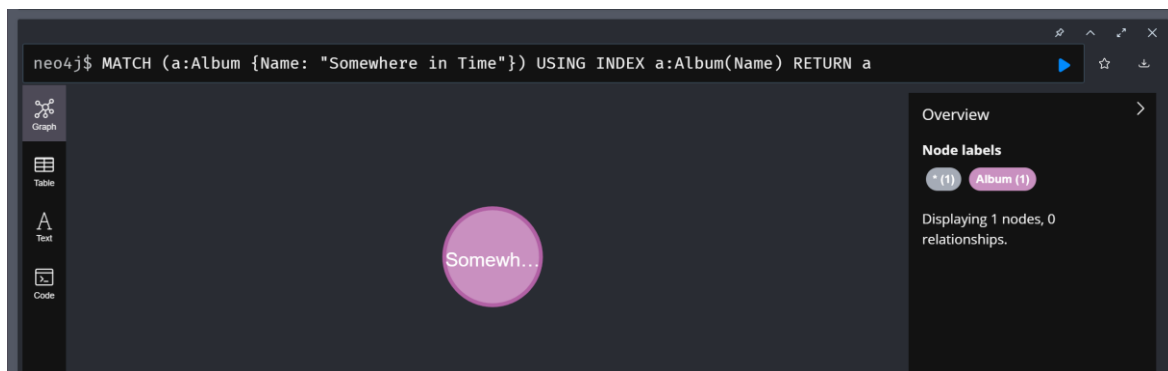
Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
index_343aff4e	LOOKUP		NODE	null	null	ONLINE
index_67e06f58	RANGE		NODE	["Album"]	["Name"]	ONLINE
index_f7700477	LOOKUP		RELATIONSHIP	null	null	ONLINE

Constraint Name	Type	EntityType	LabelsOrTypes	Properties
None				

Execute the following command to visualize what's related, and how

CALL db.schema.visualization

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.

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
constraint_bb37e9f5	RANGE		NODE	["Artist"]	["Name"]	ONLINE
index_343aff4e	LOOKUP		NODE	null	null	ONLINE
index_67e06f58	RANGE		NODE	["Album"]	["Name"]	ONLINE
index_f7700477	LOOKUP		RELATIONSHIP	null	null	ONLINE
Constraint Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
constraint_bb37e9f5	UNIQUENESS		NODE	["Artist"]	["Name"]	

حال آن را با اضافه کردن آر تیست با نام تکراری تست می کنیم:

```
neo4j$ CREATE (a:Artist {Name: "Joe Satriani"}) RETURN a
```

ERROR Neo.ClientError.Schema.ConstraintValidationFailed

Node(7) already exists with label 'Artist' and property 'Name' = 'Joe Satriani'

```
neo4j$ CREATE (a:Artist {Name: "Joe Satriani"}) RETURN a
```

Overview

Node labels

بار اول اجرا می شود و بار دوم ConstraintValidationFailed می دهد.

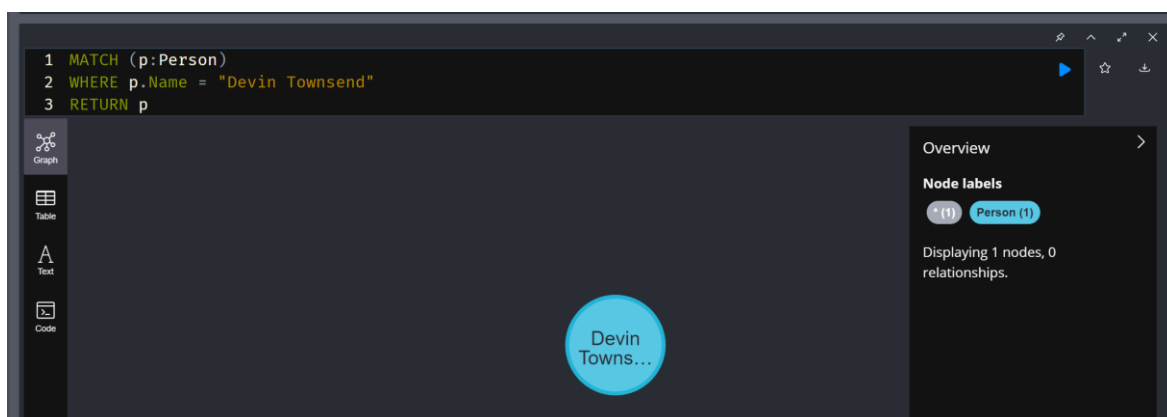
حال یک property existence constraint می سازیم.

```
neo4j$ CREATE CONSTRAINT FOR (a:Artist) REQUIRE a.Name IS NOT NULL
```

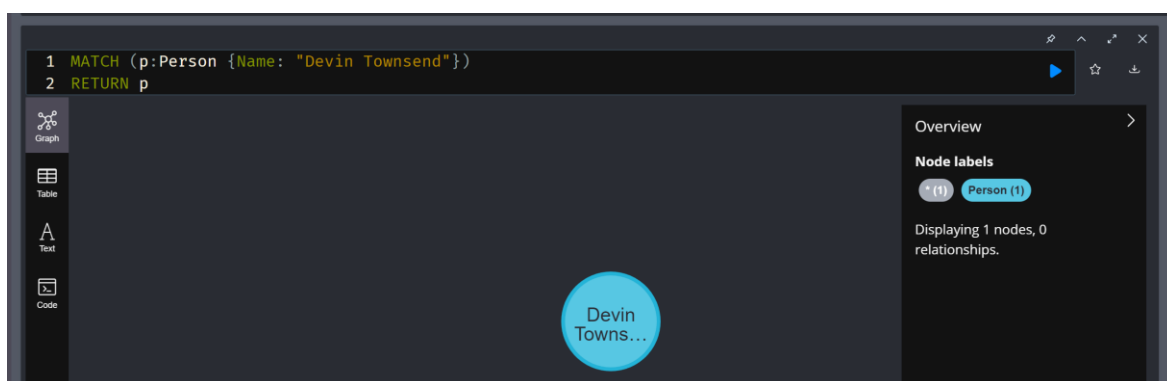
Added 1 constraint, completed after 51 ms.

Table

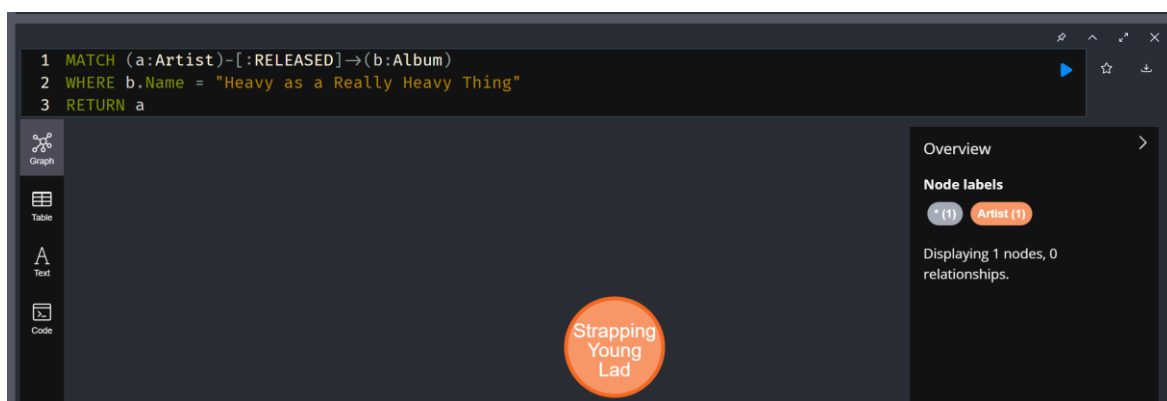
8-1. انتخاب با MATCH



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

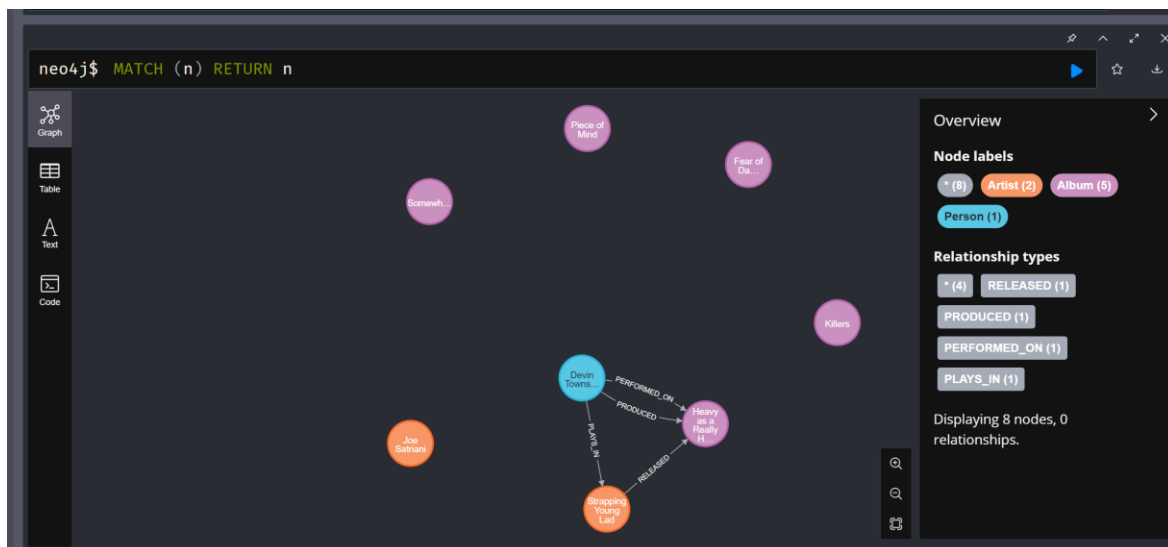


می توان علاوه بر node-ها، relationship-ها هم MATCH کرد.

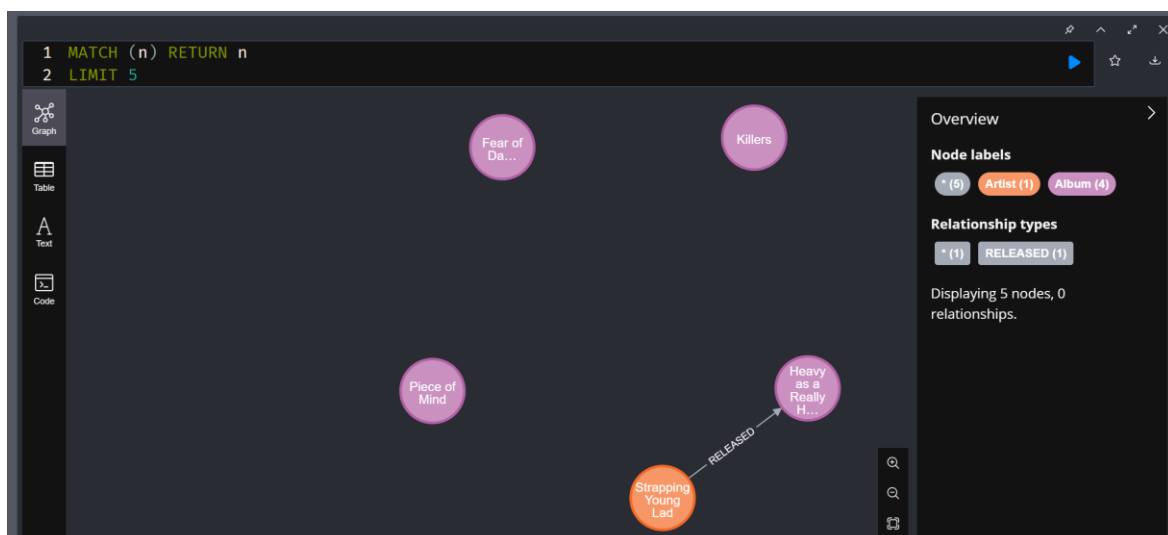


از شکل cypher می توان مفهوم کار انجام شده را به راحتی دید.

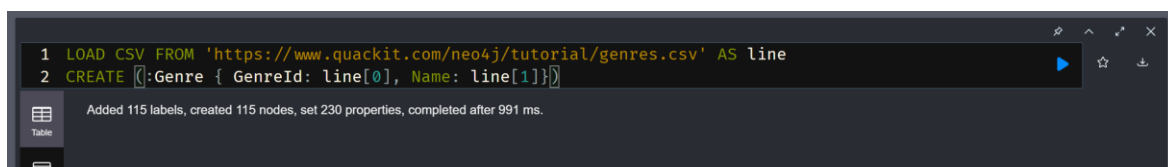
با دستور زیر می توان همه node-ها را دید:



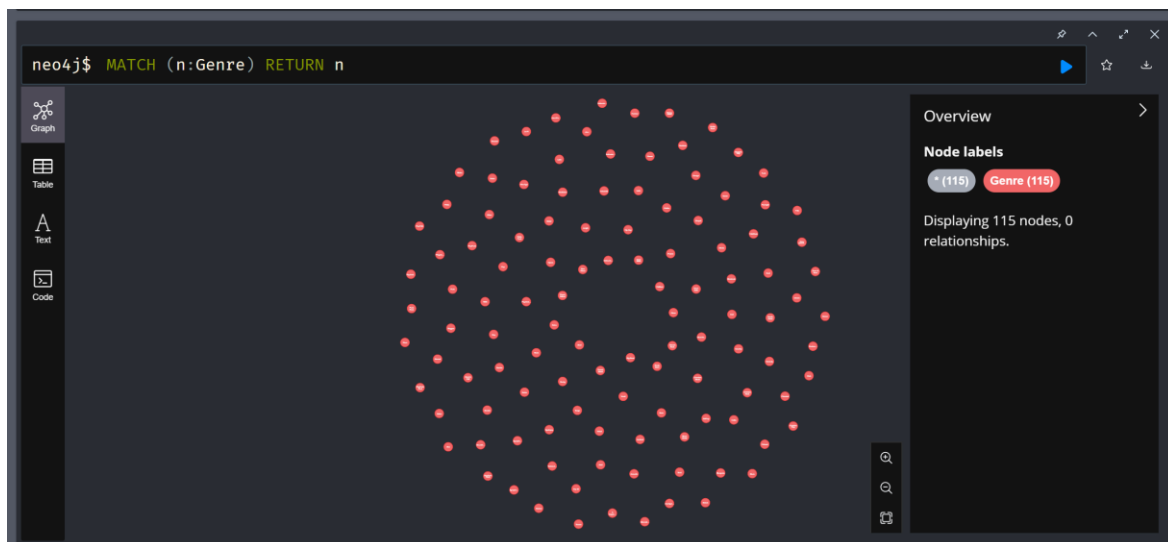
می‌شود خروجی را لیمیت کرد:



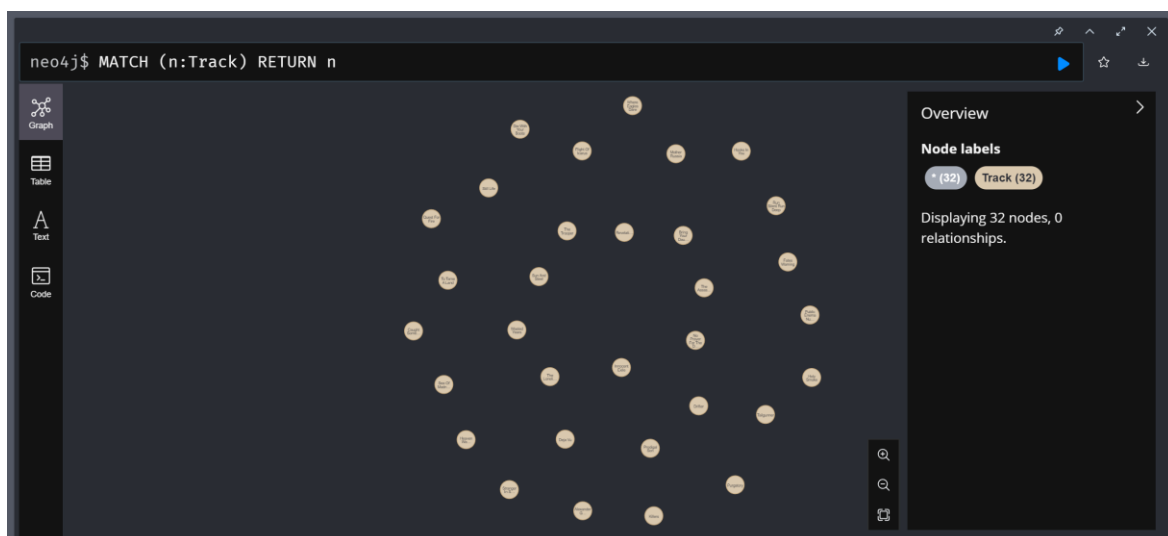
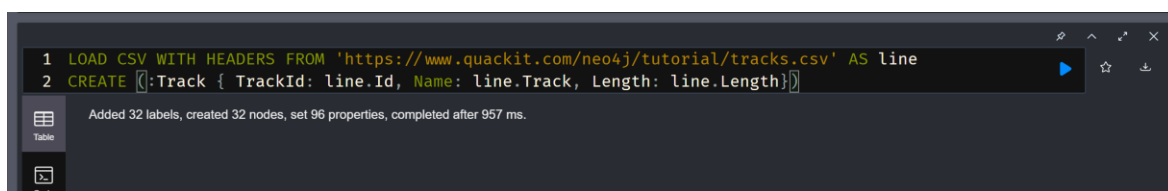
1-9. ایمپورت داده از CSV



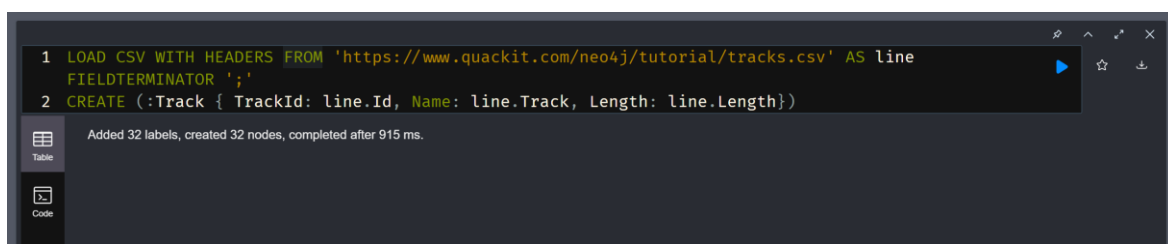
برای لود از فایل لوکال باید از [file:///](#) استفاده کرد.



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:

```

1 :auto LOAD CSV WITH HEADERS FROM 'https://www.quackit.com/neo4j/tutorial/tracks.csv' AS line
2 CALL {
3   WITH line
4   CREATE (:Track {TrackId: line.Id, Name: line.Track, Length: line.Length})
5 } IN TRANSACTIONS

```

Added 32 labels, created 32 nodes, set 96 properties, completed after 478 ms.

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

10-1. حذف Index

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

\$:schema

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
constraint_bb37e9f5	RANGE		NODE	["Artist"]	["Name"]	ONLINE
index_343aff4e	LOOKUP		NODE	null	null	ONLINE
index_67e06f58	RANGE		NODE	["Album"]	["Name"]	ONLINE
index_f7700477	LOOKUP		RELATIONSHIP	null	null	ONLINE

دنبال ایندکس روی نام آلبوم هستیم پس index_67e06f58 را می خواهیم:

```

neo4j$ DROP INDEX index_67e06f58

```

Removed 1 index, completed after 1 ms.

و می بینیم که حذف شده است:

\$:schema

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
constraint_bb37e9f5	RANGE		NODE	["Artist"]	["Name"]	ONLINE
index_343aff4e	LOOKUP		NODE	null	null	ONLINE
index_f7700477	LOOKUP		RELATIONSHIP	null	null	ONLINE

11-1. حذف Constraint

مانند حذف Index، از نام constraint استفاده می کنیم.

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.

The screenshot shows the Neo4j Cypher Shell interface. At the top, there is a table with constraint information. Below it, a text box contains the command: `CALL db.schema.visualization`. The main command area shows the execution of `neo4j$ DROP CONSTRAINT constraint_bb37e9f5`. The result indicates that 1 constraint was removed successfully after 8 ms.

12-1. حذف Node

آلبوم‌ها با نام Killers را حذف می‌کنیم.

The screenshot shows the Neo4j Cypher Shell interface. The command `neo4j$ MATCH (a:Album {Name: "Killers"}) DELETE a` is entered. The result shows that 1 node was deleted successfully after 2 ms.

حذف چند node با هم:

The screenshot shows the Neo4j Cypher Shell interface. The command `1 MATCH (a:Artist {Name: "Iron Maiden"}), (b:Album {Name: "Powerslave"})
2 DELETE a, b` is entered. The result shows that no changes or records were affected.

حذف همه node-ها:

The screenshot shows the Neo4j Cypher Shell interface. The command `neo4j$ MATCH (n) DELETE n` is entered. An error message is displayed: `Neo.ClientError.Schema.ConstraintValidationFailed`. The message states: "Cannot delete node<0>, because it still has relationships. To delete this node, you must first delete its relationships."

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.