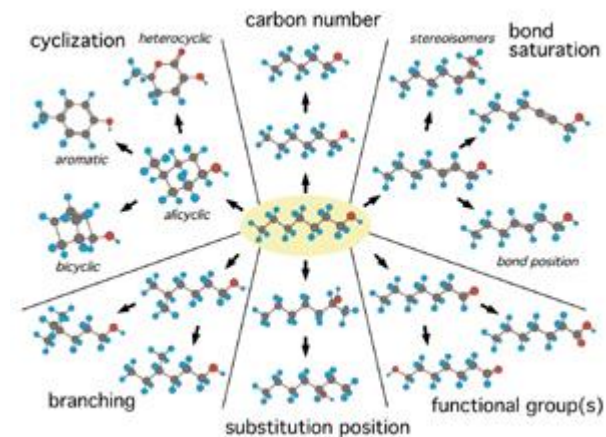
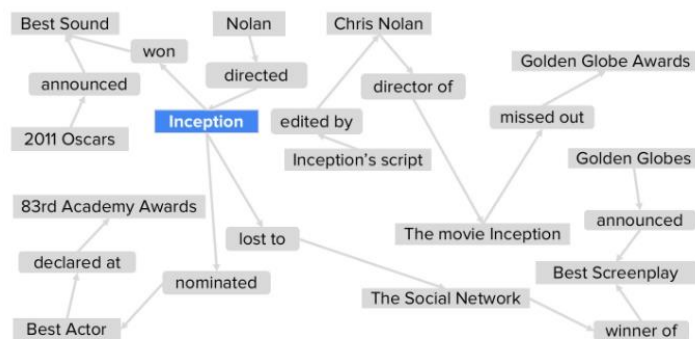


- Graph DB 부각 이유

소셜 네트워크, 생물학, 화학, 비즈니스 등 다양한 분야에서 그래프(Graph) 구조의 데이터가 활발히 생성

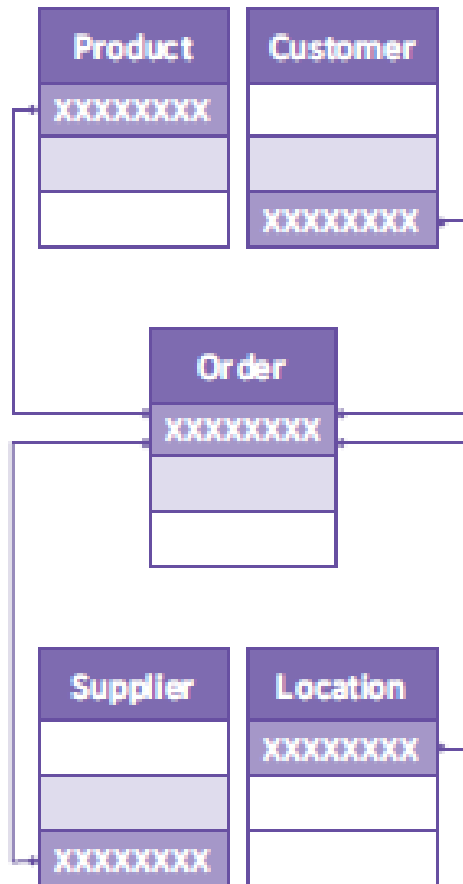
지식 그래프, 소셜 네트워크에서 사람들 간의 관계, 화학 분야에서 여러 원소가 결합된 복잡한 화합물  
비즈니스에서 각 데이터에 대한 이력 정보(유통, 거래, 배포 기록 등)



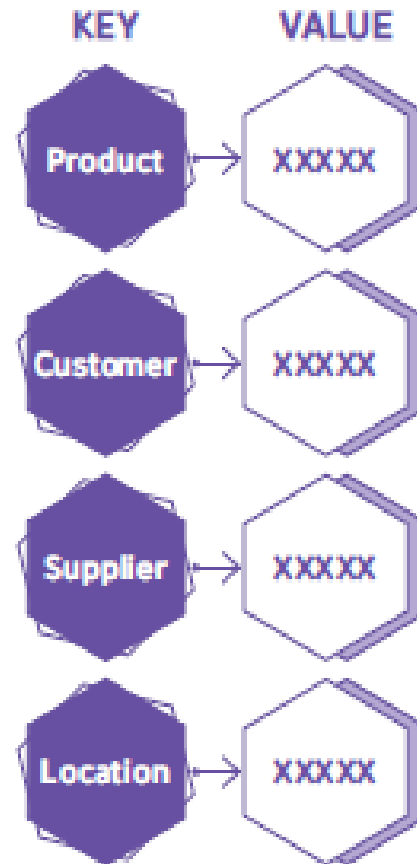
기존의 관계형 데이터베이스를 사용하는 경우, 저장에 매우 복잡, 다수의 테이블 간의 조인으로 비효율  
Key-Value 기반, Document 기반, Column 기반 등의 NoSQL 데이터베이스에서도 동일한 문제 발생

그래프 형태 데이터의 효율적인 저장 및 질의를 위해 그래프 데이터베이스가 활발히 사용

Relational DB



key-value DB



Graph DB

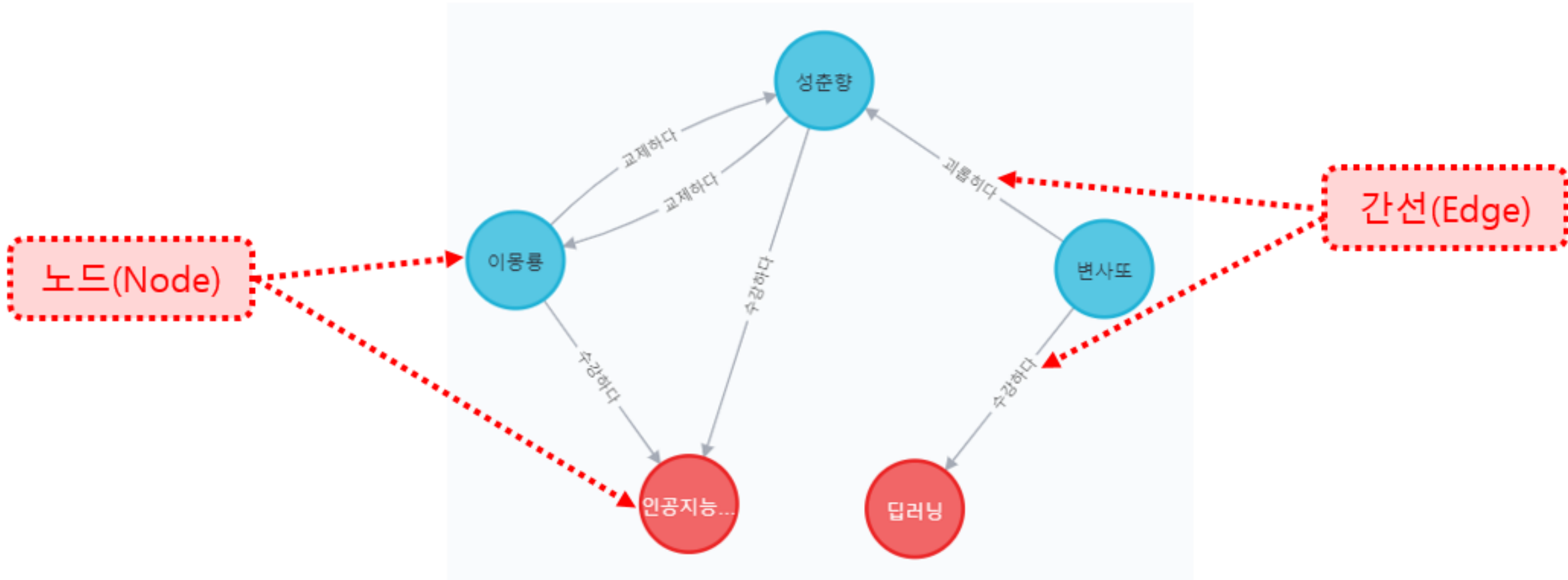


- Graph DB란?

그래프 구조의 데이터를 효율적으로 저장하고 관리하기 위한 데이터베이스 관리 시스템

스키마(Schema)가 존재하지 않으며, 직관적인 모델링 가능

노드(Node)와 간선(Edge)로 구성되며 Edge를 통해 Node와 Node간의 관계(relationship)를 저장



- Neo4j

고유의 Graph 모델을 제안하고 구현한 최초의 Graph DB로 가장 많이 사용

2007년에 처음 배포를 시작하였으며, 오픈소스로 제공

관계형 데이터베이스 등을 사용하지 않고 그래프의 저장 및 질의를 직접 처리

Graph 데이터에 대한 질의 언어로 Cypher라는 질의 언어 제공

데이터 접근을 위해 Java, Neo4j-OGM, RESTful HTTP API, Spring Data Neo4j 등 다양한 API를 제공

Java, JavaScript, PHP, Python, Ruby, Scala, Go 등 다양한 프로그래밍 언어 지원

- GraphDB

GraphDB는 그래프와 RDF 데이터를 저장하고 질의하기 위한 그래프 데이터베이스로서, 그래프에 대한 추론 및 클러스터링 기능도 추가로 제공한다.

Java로 구현되었으며, 그래프 외에도 OWL/RDFS 스키마 형태의 데이터를 저장 가능  
데이터에 대한 접근을 위해 GeoSPARQL, RDF4J, API, Java API, Sesame REST HTTP 프로토콜 등을 지원  
C#, Clojure, Java, JavaScript, PHP, Python, Ruby, Scala등의 프로그래밍 언어 지원을 지원  
상업용 버전 외에도 GraphDB-Free라는 무료 버전을 제공

- Dgraph

대규모 그래프에 대한 분산 저장 및 처리를 지원하는 Graph DB  
그래프에 특화된 저장소를 사용하며 Go언어를 사용하여 구현  
2016년에 처음 배포되었으며, 현재 Apache 2.0 라이선스에 따라 오픈소스로 제공  
질의 언어로는 JSON 형태의 GraphQL을 제공하며, gRPC API, HTTP API를 제공  
Go, Java, C#, C++, JavaScript, PHP, Python, Ruby 등의 프로그래밍 언어 지원

- TigerGraph

대규모 그래프에 대한 고속의 분석 지원을 목표로 개발된 분산 및 병렬 그래프 처리 플랫폼

2017년에 처음 배포되었으며, 현재 상업용 제품으로 판매 중

C++로 구현되었으며, 현재 Linux에서만 실행 가능

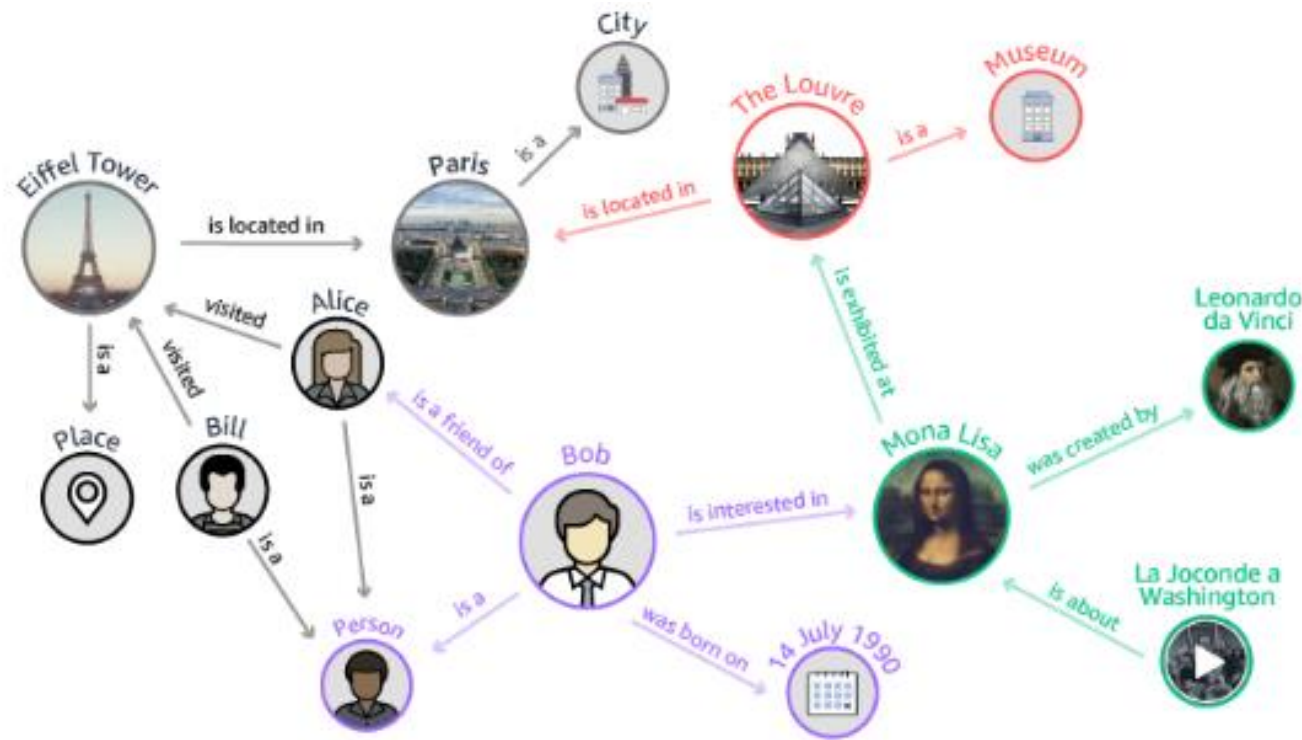
질의 언어로는 SQL을 그래프 데이터에 맞게 확장한 GSQL을 제공, Kafka, RESTful HTTP/JSON AP 등도 제공

C++과 Java 등의 프로그래밍 언어 지원

- 지식 그래프(Knowledge Graph)

지식 그래프를 사용하면 정보를 그래프 모델에 저장

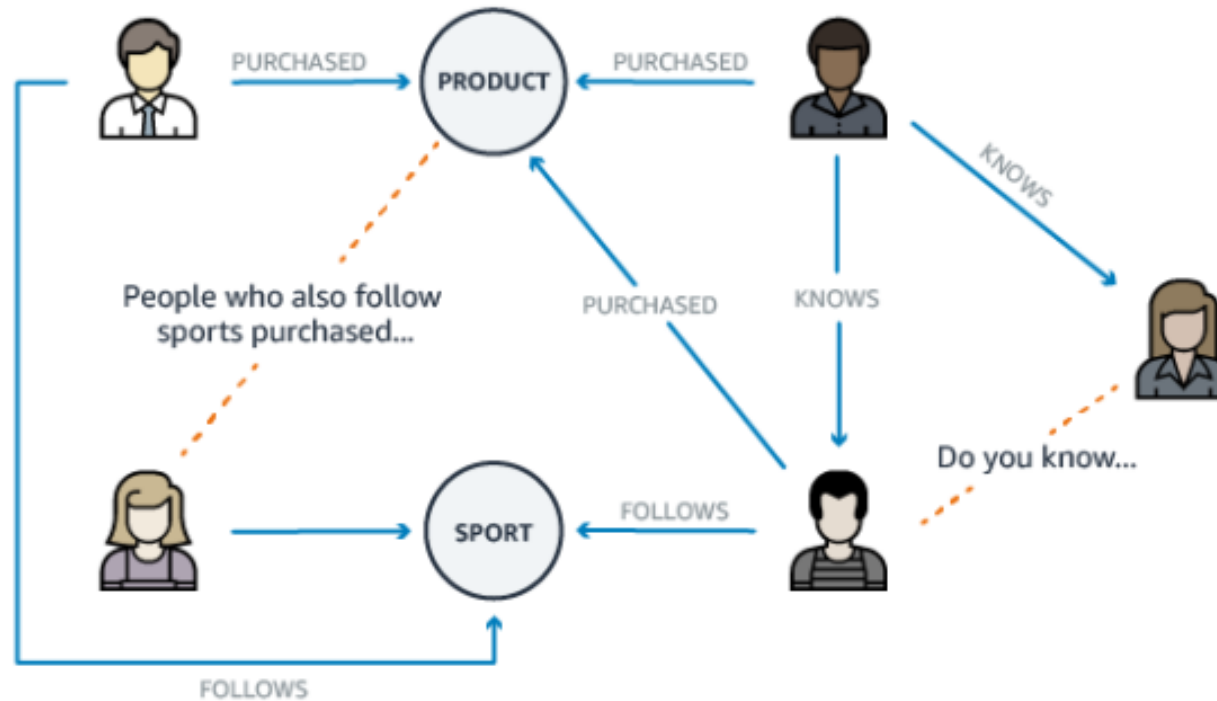
질의 언어를 이용하여 데이터 간 상호연결성이 높은 질의결과 조회 가능





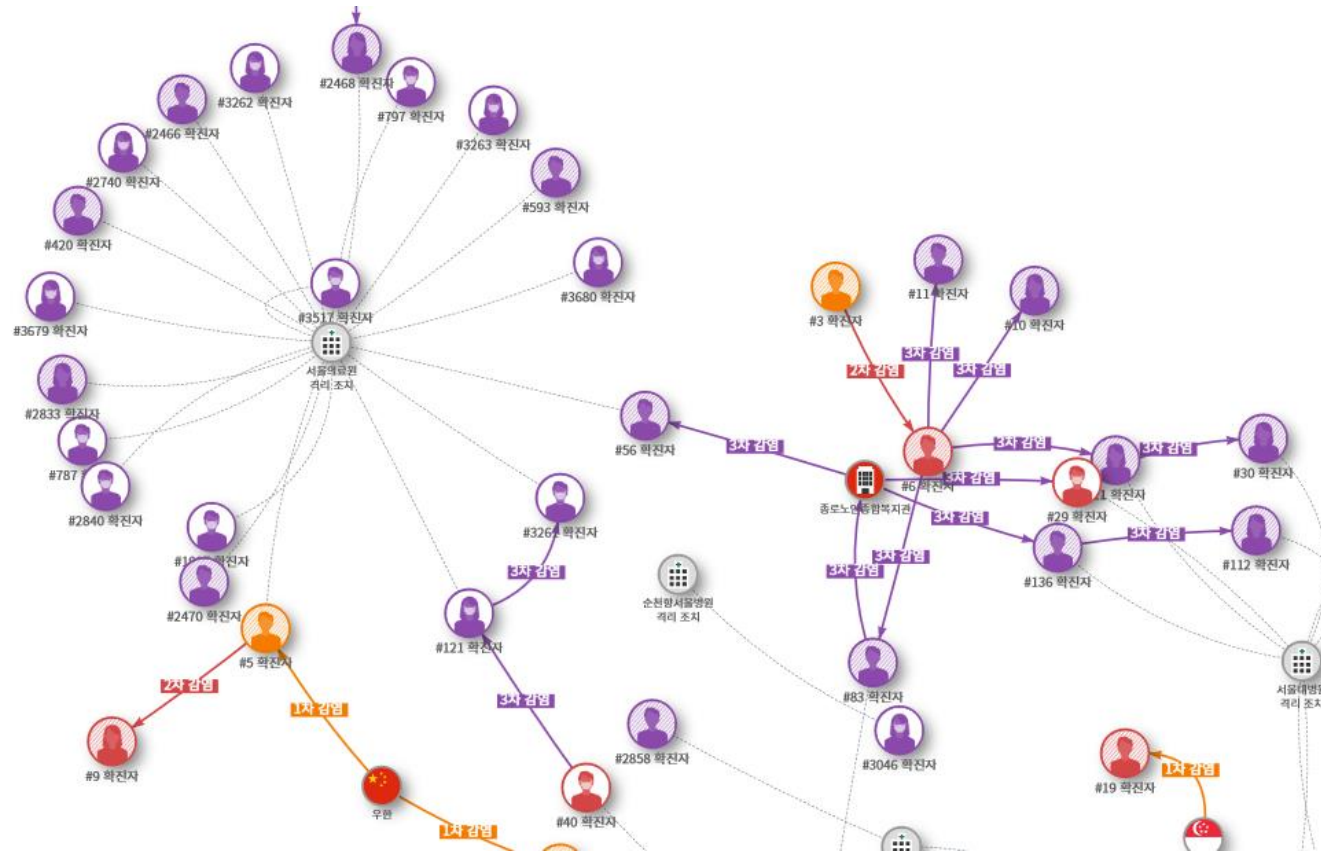
- 추천 시스템(Recommend System)

고객 관심사, 친구, 구매 이력과 같은 데이터 간의 관계를 그래프로 저장  
데이터 간의 관계를 질의하여 맞춤형 개인화 서비스 제공 가능



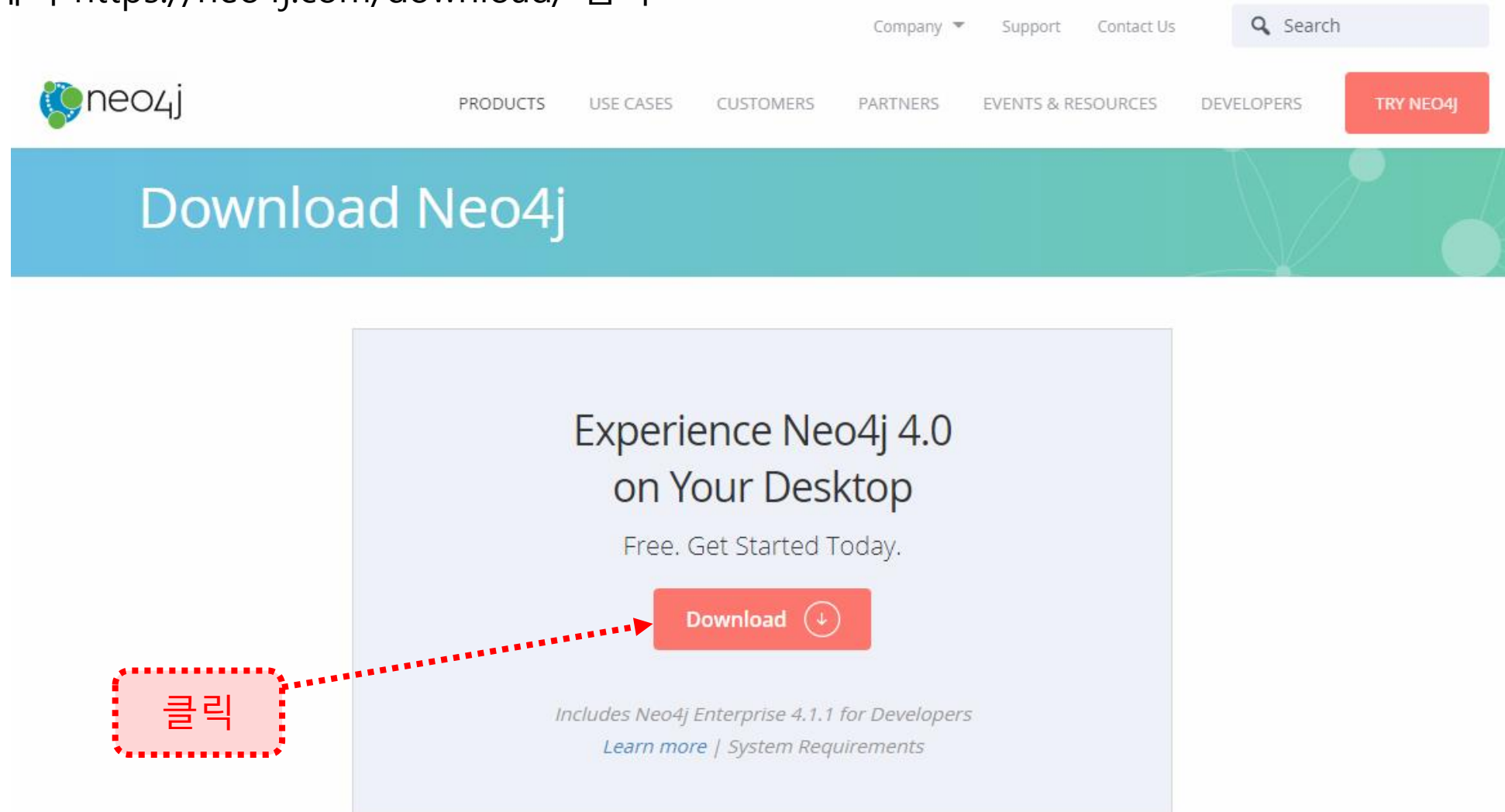
- 코로나 확산·감염 경로 추적

코로나 확산·감염 경로와 감염자 간의 접촉 관례를 저장  
질의를 통해 감염 경로 추적 및 예측에 활용 가능



- Neo4j 다운로드

브라우저에서 <https://neo4j.com/download/> 접속



- Neo4j 다운로드  
개인정보 입력 후 다운로드

Get Neo4j Desktop

## Get Started Now

Please fill out this form to begin your download

\*

\*

\*

\*

\*

By downloading you agree to the [Neo4j License Agreement for Neo4j Desktop Software](#).

Download Desktop

The information you provide will be used in accordance with the terms of our [privacy policy](#).

정보 입력

클릭

- Neo4j 다운로드

# Neo4j Desktop Activation Key 복사

Thanks for downloading Neo4j Desktop

Your download should begin automatically in a few seconds. If it doesn't, Click one of the links : [Windows](#) • [OSX](#) • [Linux](#)

**Recommended system requirements:** MacOS 10.10 (Yosemite)+, Windows 7+, Ubuntu 12.04+, Fedora 21, Debian 8.

## Neo4j Desktop Activation Key

Use this key to activate your copy of Neo4j Desktop for use.



eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJjYXVsYmFjayI6IlIsImVtYWlsIjoieYmpiraW0yMDA0QGdtYWlsLmNvbSIsImZvcmlhdCI6Impzb24iLCJvcnciOiJxb29yaUZlYyIsbnB1YiI6Im5lbzRqLmNvbSIsInJlZyI6IkJP

SyBKVSBSLU0iLCJzdWIIoiJuZW80ai1kZXNrdG9wiwiZXhwIjoxNjI2NjE4MzM0LCJ2ZXIIoiIqlwiaXNZljoib

mVvNGouY29tliwbmJmljoxNTk1MDgyMzM0LCJpYXQiOiEOTUwODIzMzQsimp0aSjI6BZRWxSZklntij

9.FnjoSqN\_9-j7OJBKfGXn5eGTWcmjxkCmRgNXtn3u0Af\_RAC29FfkwkM4UMESmLIuPWlyhcW4dHLW

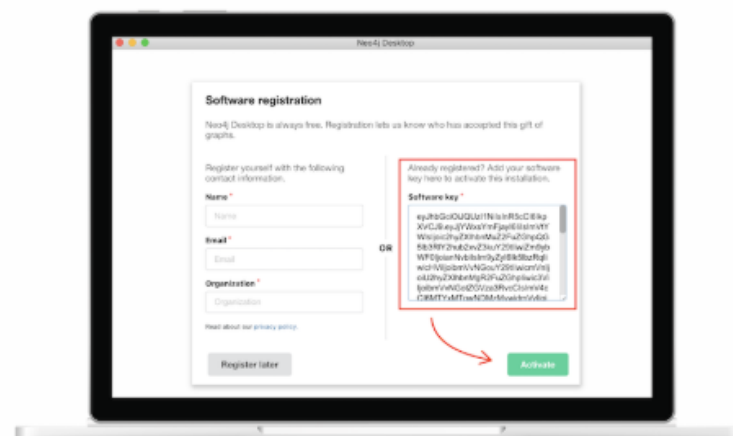
MGXoUEyLyjY6ULl8z-KZ2IDztRgr-TjYu\_WoeE-qcM1K1u\_hP8nxesQWSI9hWkqqI245dbK9GrblP4nM

rxv4JVbC7mO4pwZq1rlWbf47KF9nvduUICiXo1751J3VyryErgUg1mDsiSRx2\_pwV55TtMiib4sqUA2VtdB

2ba\_W43hAtuOaeYvgdr2JG8rbyMt6goizoXGN0G4HAb6v4cyKSUHLSOuifypRwZYIPy6zv2EE-zYAgS2TBX

CVI9R3dnEioek9jq

복사



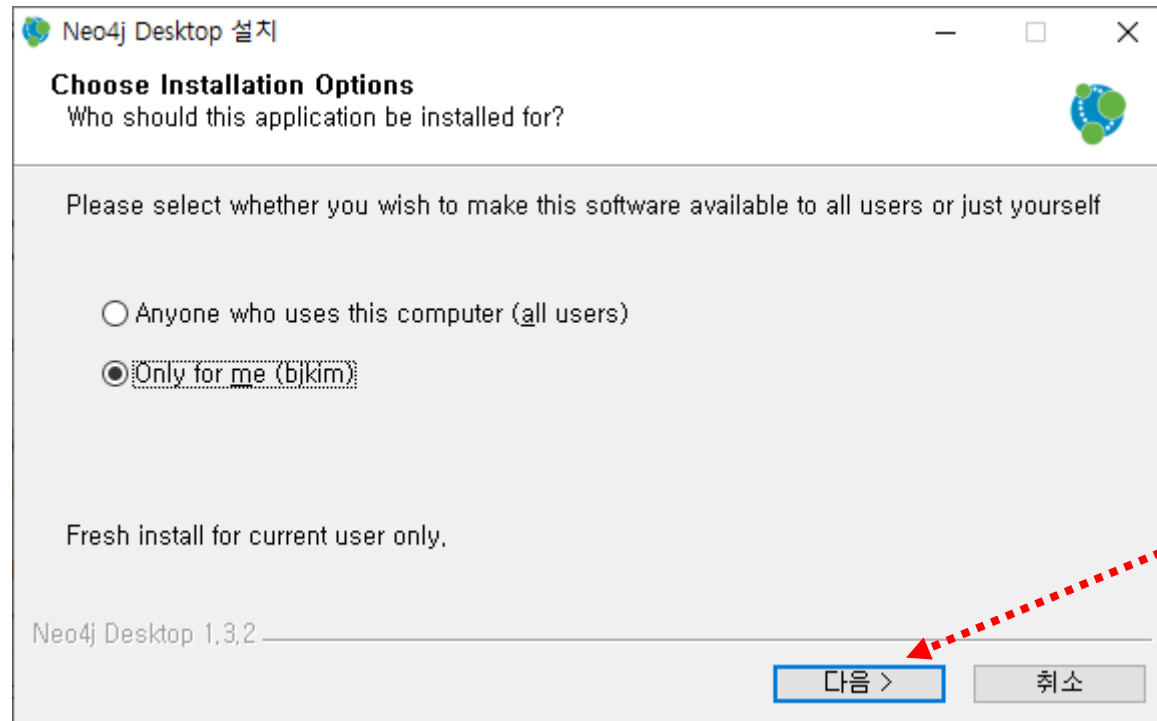
## Installation Video



## Installation Guide

- Neo4j 설치

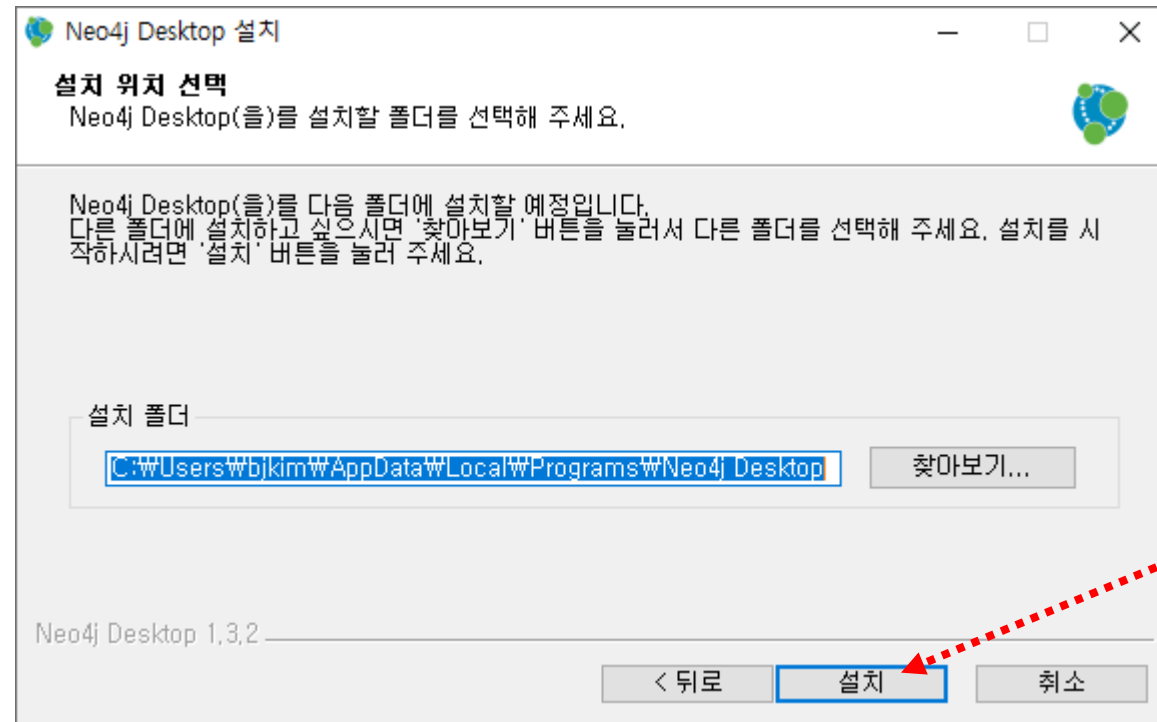
해당 계정에만 사용가능한 Only for me 옵션 선택



클릭

## Neo4j 설치

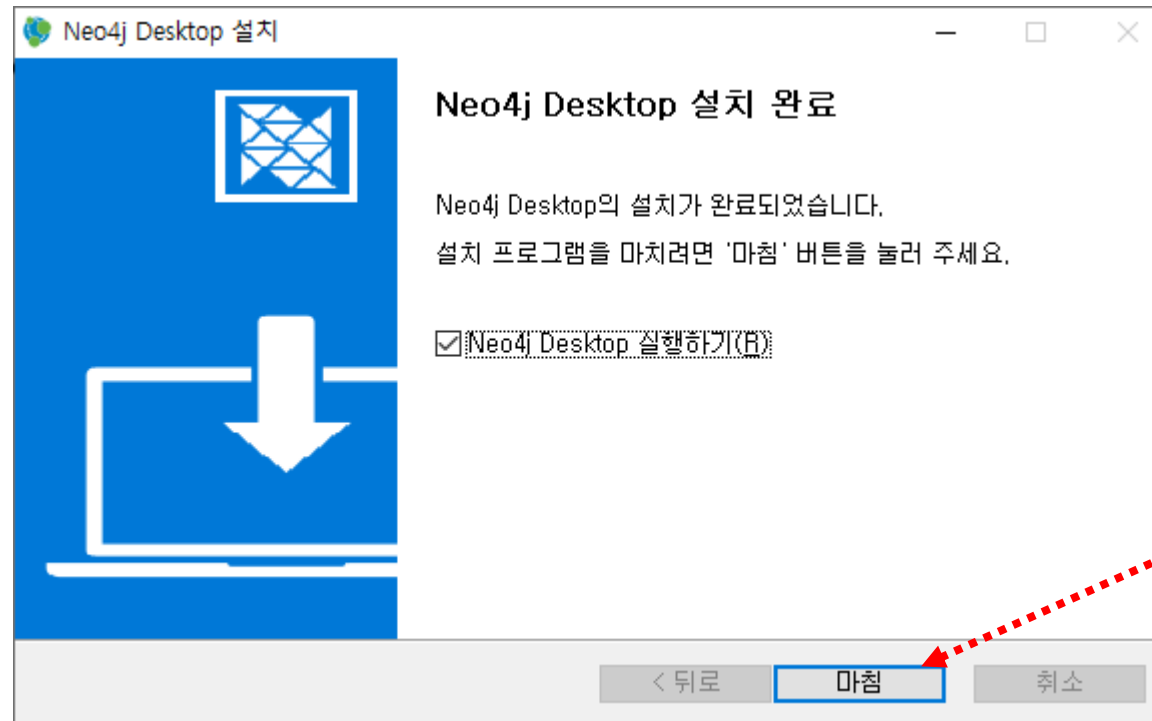
Neo4j의 설치 디렉토리 확인 필요



클릭

- Neo4j 설치

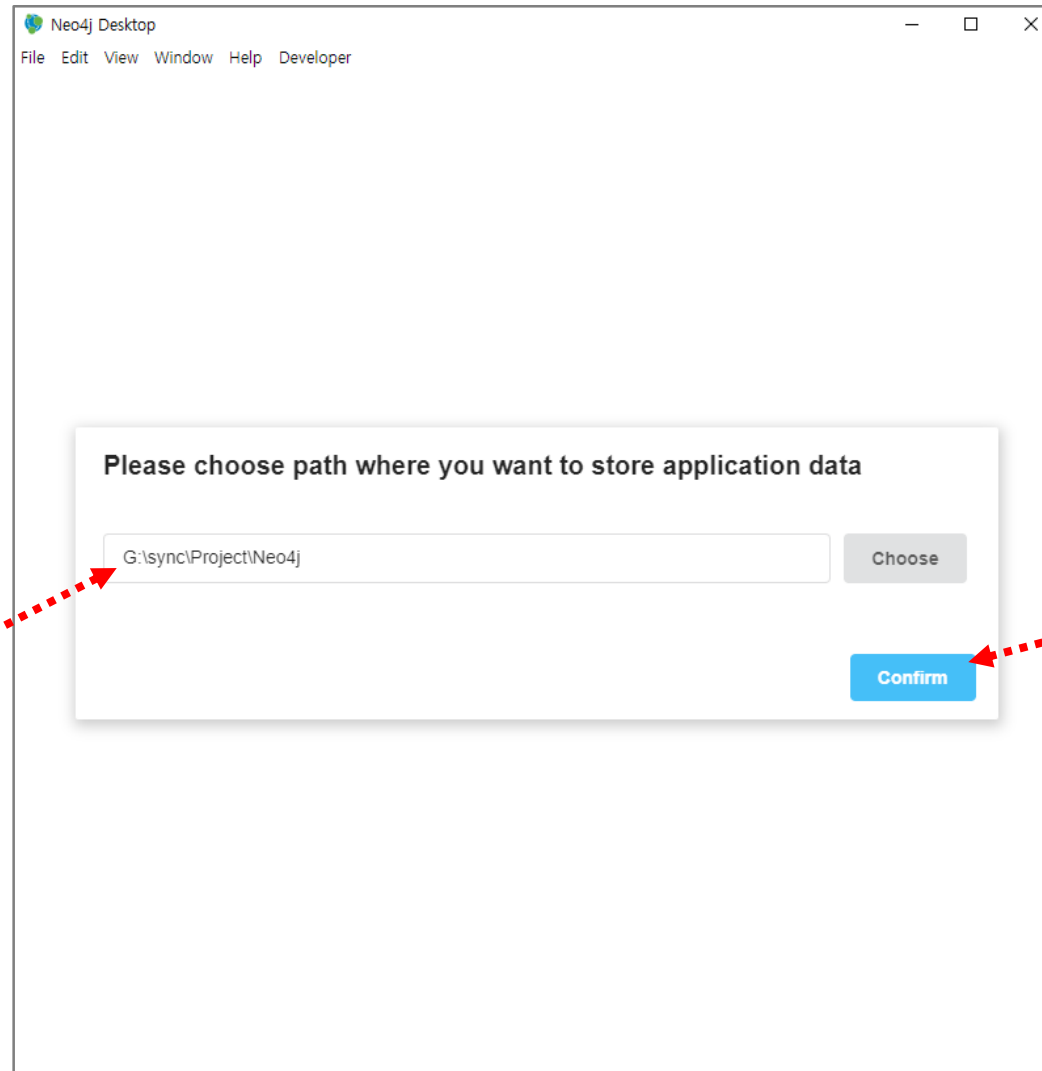
설치 종료 및 Neo4j 실행





- Neo4j 실행

저장소 디렉토리 선택 및 확인



저장 디렉토리 선택

클릭

## Neo4j 정보 입력 및 Key 복사

Neo4j Desktop

File Edit View Window Help Developer

### Software registration

Neo4j Desktop is always free. Registration lets us know who has accepted this gift of graphs.

Register yourself with the following contact information.

Name \*  
BOK JU KIM

Email \*  
your-email@gmail.com

Organization \*  
Woorifis

[Read about our privacy policy.](#)

OR

Already registered? Add your software key here to activate this installation.

Software key \*  
qcM1K1u\_hP8nxesQWSI9hWkq  
ql245dbK9GrbIP4nMrxv4JVbC7  
mO4pwZq1rIWbf47Kf9nvdWUI  
CiXo1751J3VyrEryUg1mDsiSRx2  
\_pwV55TtMiib4sqUA2VItdB2ba\_  
W43hAtuOaeYvgdr2JG8rbyMt6  
gOizoXGn0G4hAb6v4cyKSUHLs  
OujfYpRwZYIPy6zv2EE-  
zYAgS2TBXCVJ9R3dnEjoek9jQ

Register later

Activate

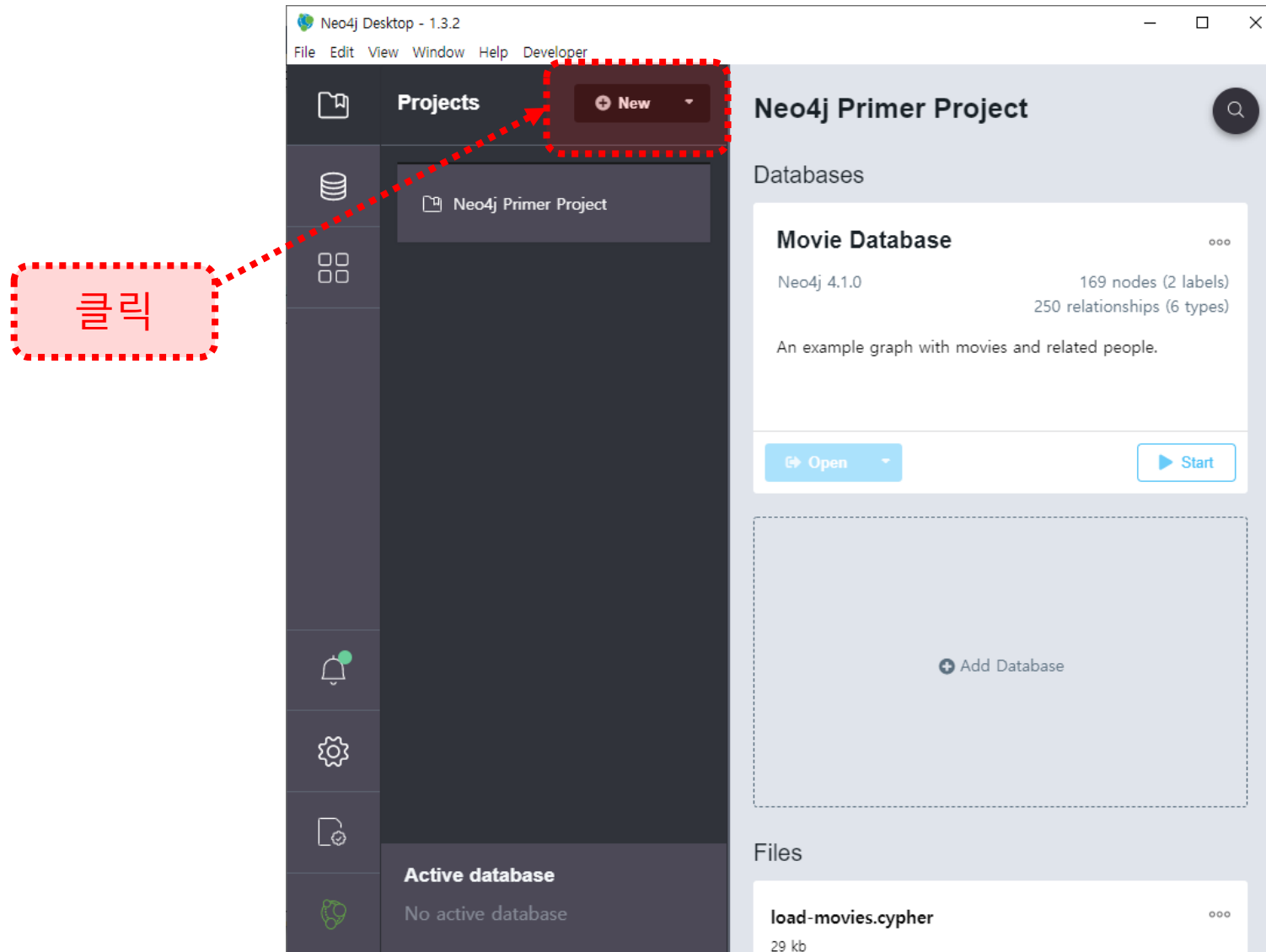
정보 입력

Key 복사

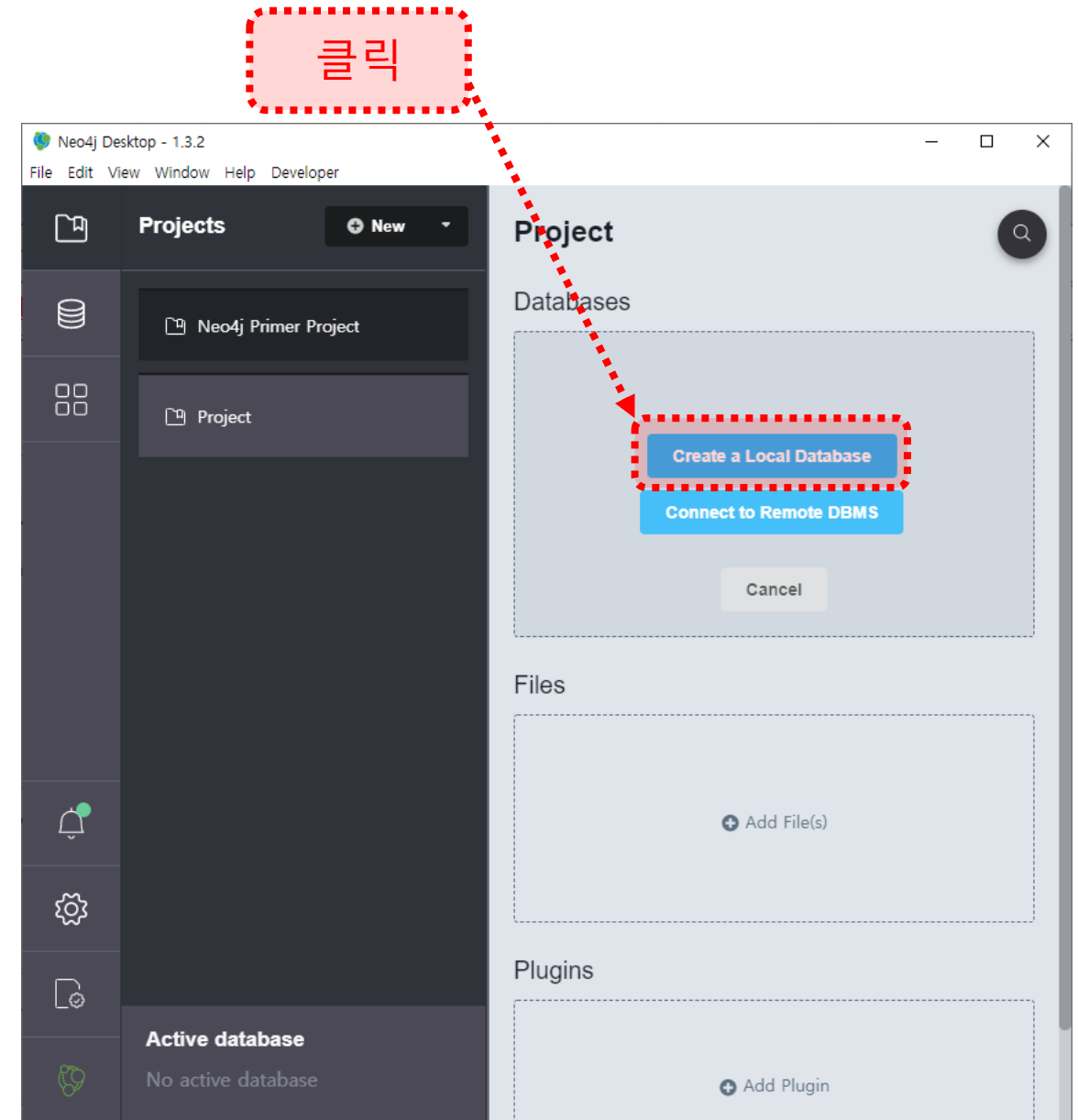
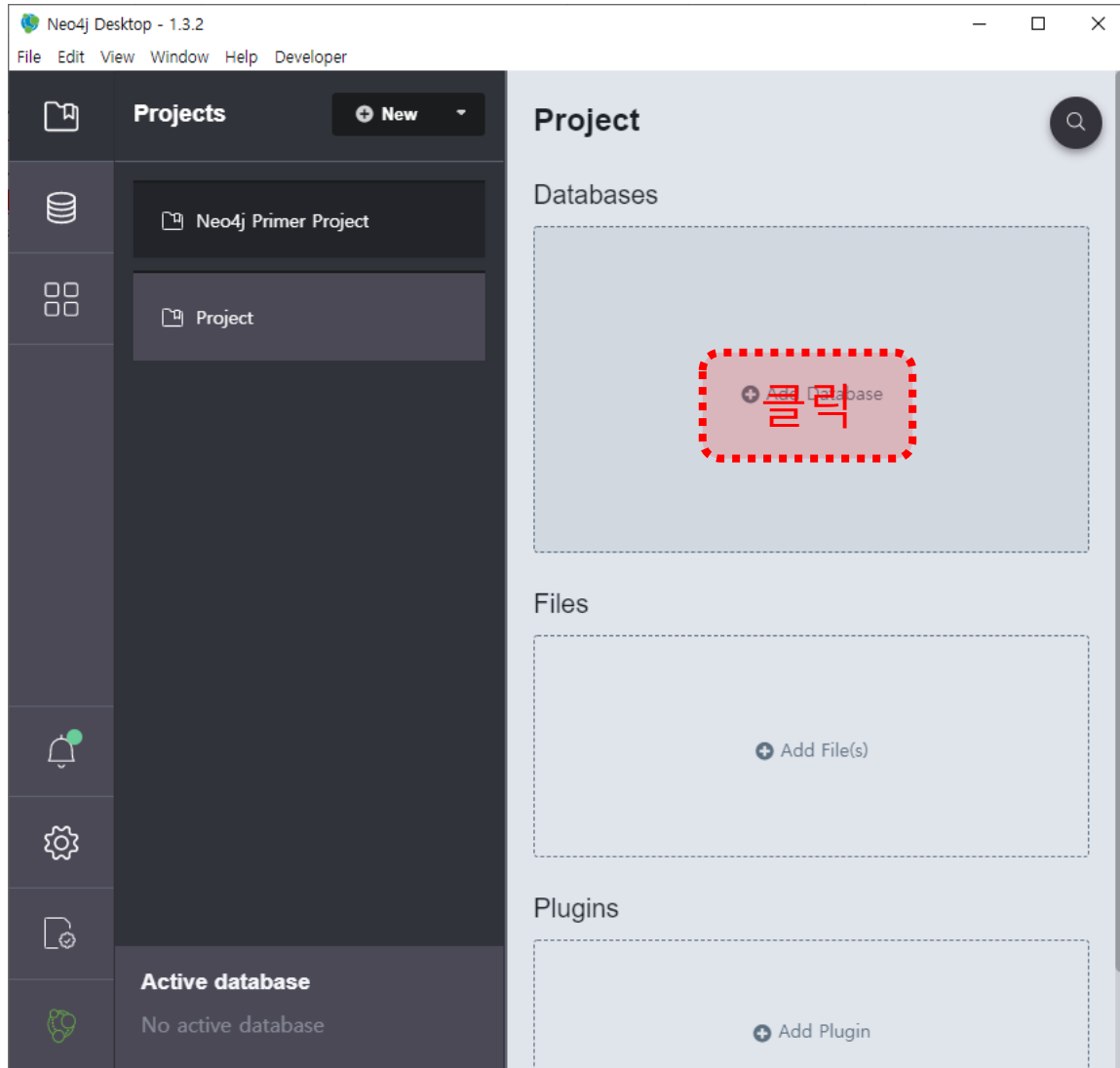
클릭

# Neo4j 다운로드 및 설치 실습

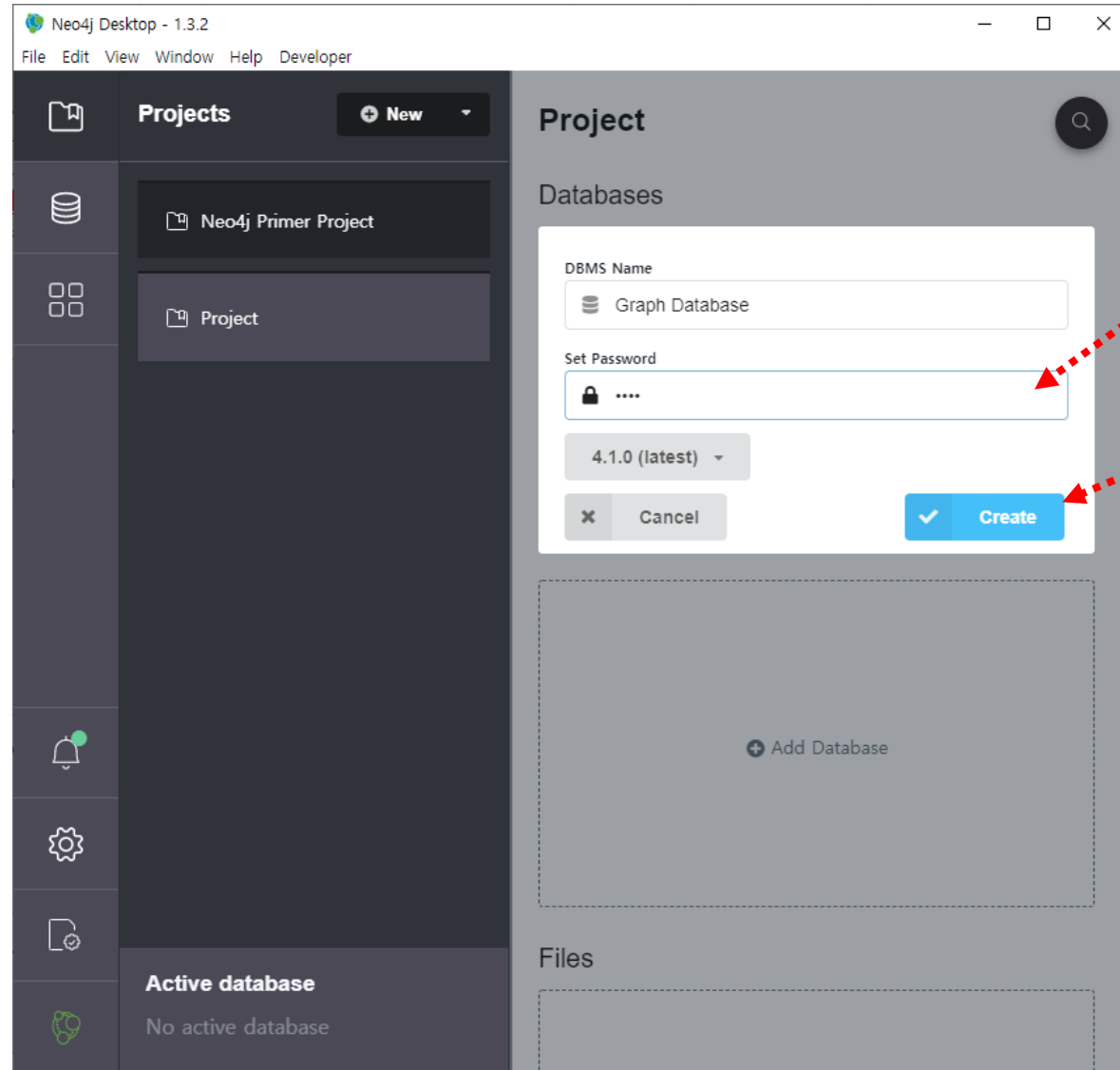
## Neo4j Project 생성



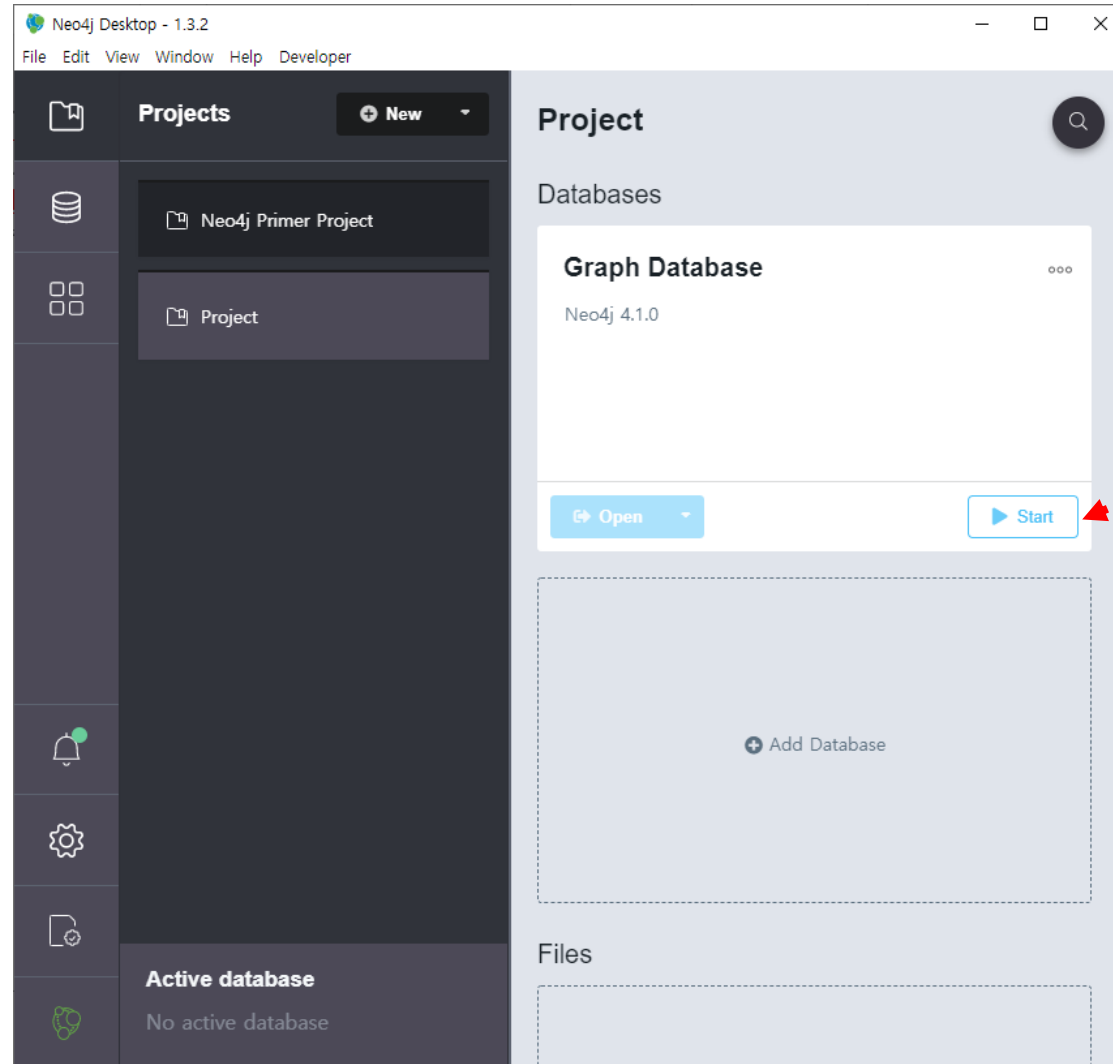
## Neo4j Database 생성



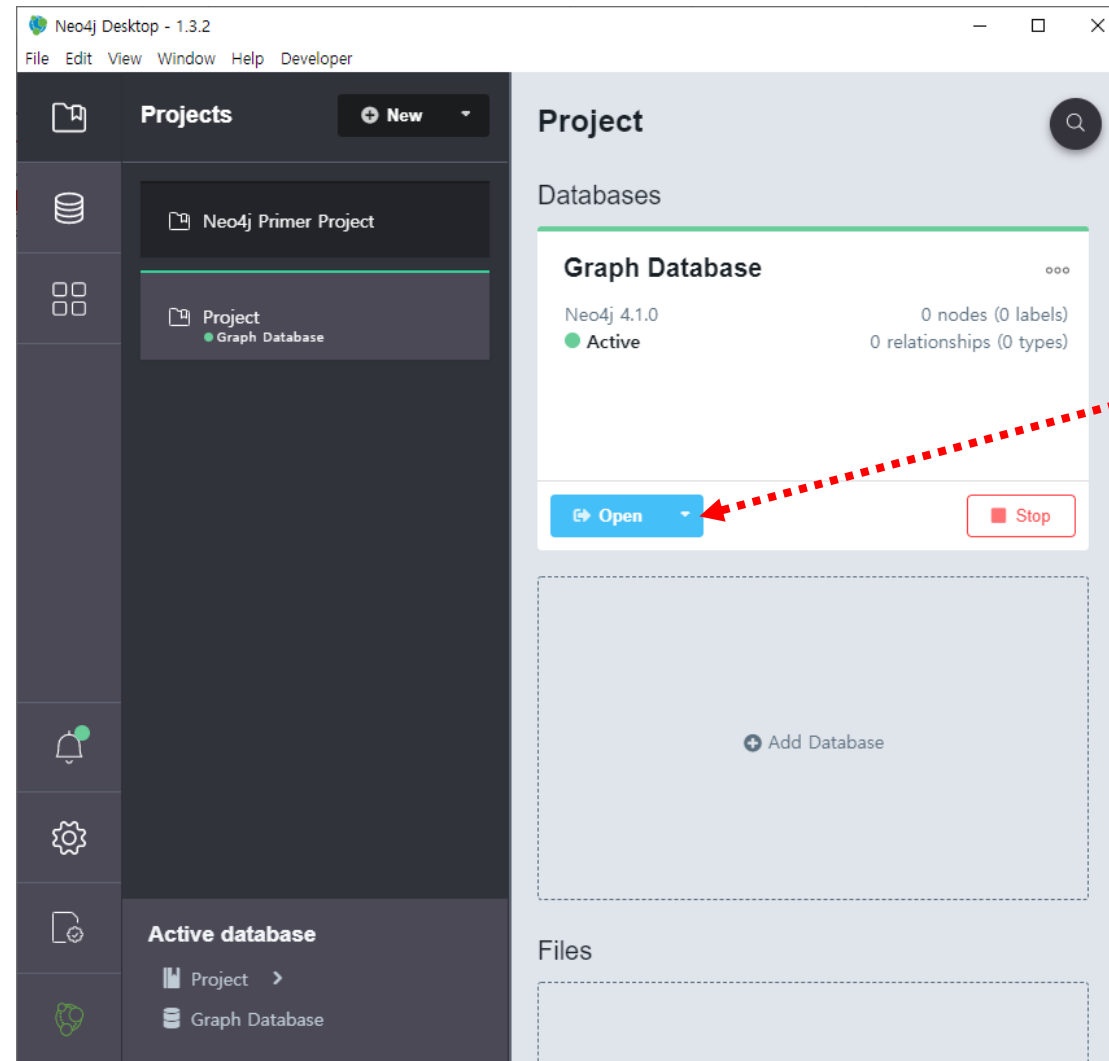
## Neo4j Database 생성



- Neo4j Database Start

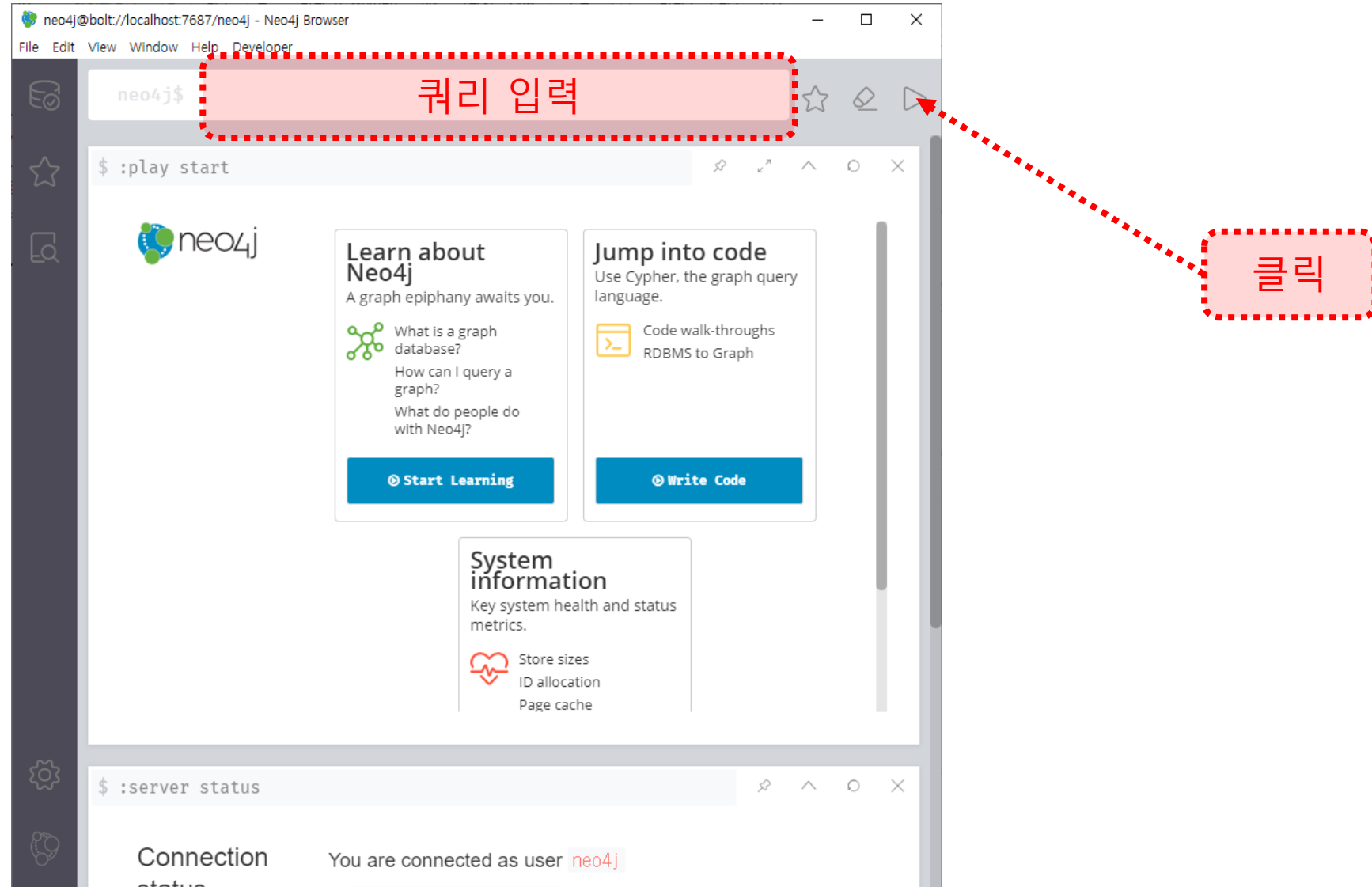


## Neo4j Database Start



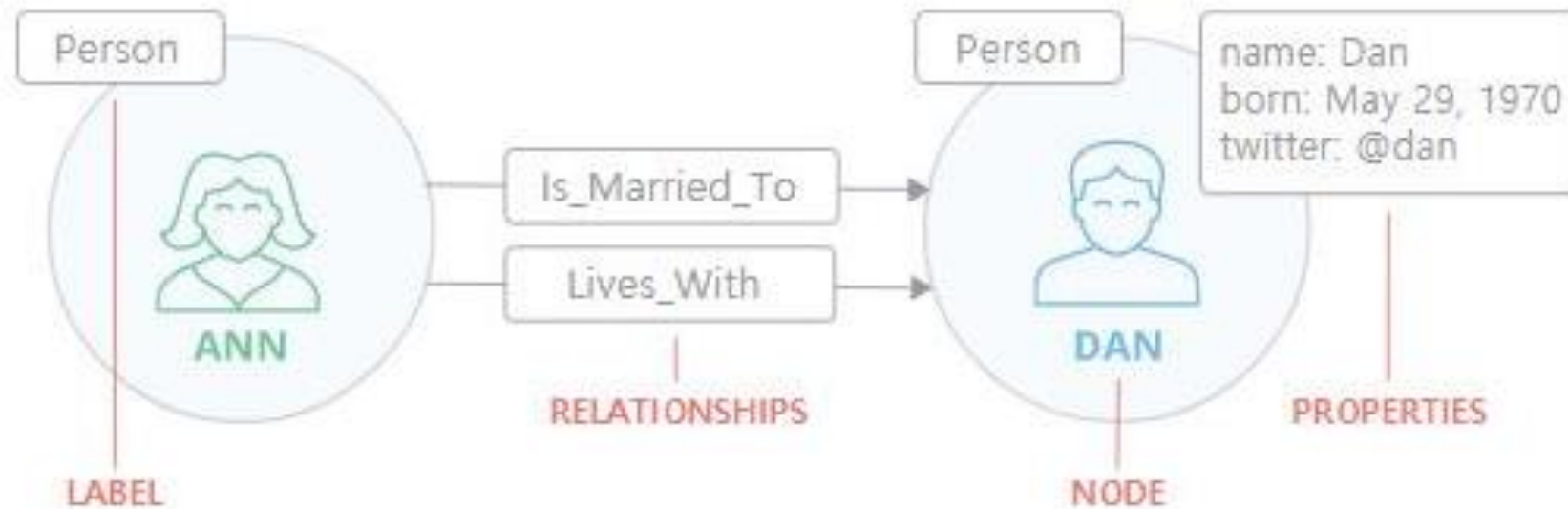


## Neo4j Database Start



- Neo4j의 데이터 구조

노드(Node)와 관계(Relationship), 노드와 관계에 대한 속성(Property), 노드의 집합인 레이블(Label) 등으로 구성

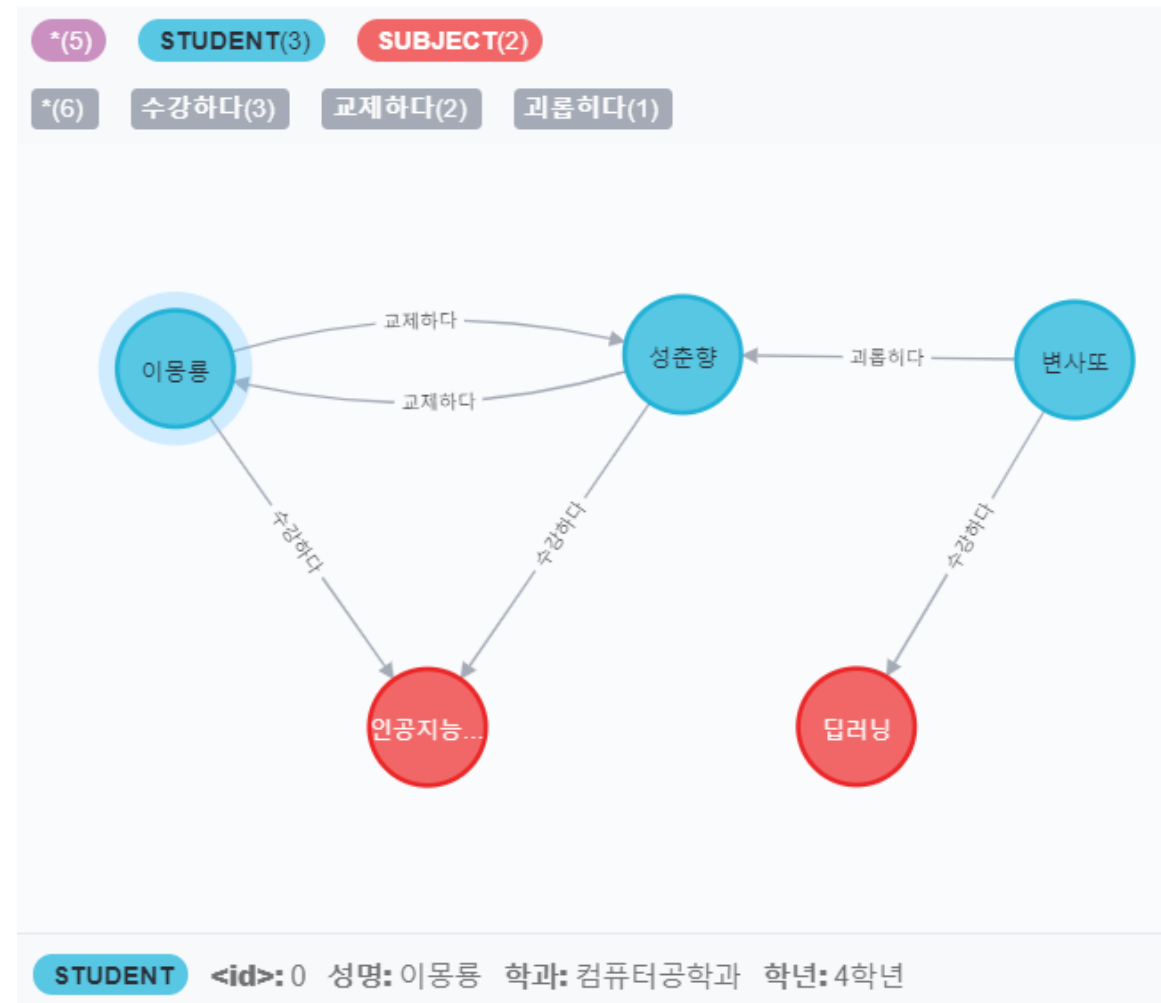


출처 : <https://neo4j.com/>

- RDB와 Neo4j의 데이터 구조 비교

RDB의 테이블은 Neo4j의 레이블에 해당, RDB의 ROW는 Neo4j의 노드에 해당

테이블명 : STUDENT		
성명	학년	학과
이몽룡	4	컴퓨터공학과
성준향	3	컴퓨터공학과
변사또	2	컴퓨터공학과
테이블명 : SUBJECT		
과목명	담당교수	학점
인공지능기초	김교수	2
딥러닝	박교수	3



# Neo4j Project 및 Database 생성 실습

### ▪ Cypher Query

Cypher Query는 Graph DB인 Neo4j의 효율적인 질의를 위한 그래프 질의어

Cypher는 처음 neo4j를 위한 질의 언어로 개발되었으나, Open Cypher 프로젝트를 통해 개방

Agens Graph를 포함한 여러 Graph DB에서 채택

Create, Match, WHERE 및 RETURN 구문이 가장 대표적인 구문

- CREATE : 노드(Node)와 관계(relationship)를 생성하기 위한 구문
- MATCH : 일치하는 그래프 패턴 검색, 그래프에서 데이터를 추출하기 위한 구문
- WHERE : MATCH 구문 내에서 조건을 추가하거나 중간 결과를 필터링하기 위한 구문
- RETURN : 결과를 반환하기 위한 구문

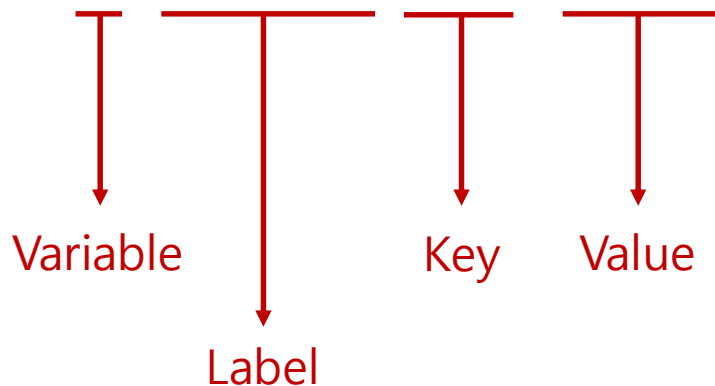
- Cypher Query CREATE - Node 생성

노드(Node)나 관계(Relationship)을 생성할 때 사용

CREATE ( Variable : Label { Key : Value, Key : Value } ) 의 형태로 작성

Key값에는 따옴표를 미사용, Value 값은 따옴표를 사용 필수(Value값이 숫자인 경우 제외)

```
CREATE ( a : STUDENT { 성명 : "이몽룡", 학년 : "4학년", 학과 : "컴퓨터공학과" } )
```



```
CREATE ( a : STUDENT { 성명 : "이몽룡", 학년 : "4학년", 학과 : "컴퓨터공학과" } ) RETURN a
```

노드 생성 후 해당 노드를 리턴하여 그래프로 출력(Variable 'a'는 해당 명령 수행 동안 임시로 사용되는 변수)

### ▪ Cypher Query CREATE - Node 생성

```
neo4j$ CREATE (a:STUDENT { 성명 : "이몽룡", 학년 : "4학년", 학과 : "컴퓨터공학과" }) RETURN a
```

neo4j\$ CREATE (a:STUDENT { 성명 : "이몽룡", 학년 : "4학년", ...

Graph

\*(1) STUDENT(1)

Table

Text

Code

이몽룡

STUDENT <id>: 0 성명: 이몽룡 학과: 컴퓨터공학과 학년: 4학년

### ▪ Cypher Query CREATE - Relationship 생성

CREATE ( a ) - [ r : 수강하다 ] -> ( b )

Node 변수      Relation 변수 Relationship      Relationship 방향      Node 변수

MATCH ( a { 성명 : '이몽룡' } ) MATCH ( b { 과목명 : '인공지능기초' } )

성명이 '이몽룡'인 Node를 검색하여 a 에 저장, 과목명이 '인공지능기초'인 Node를 검색하여 b에 저장

CREATE ( a ) -[r:수강하다] -> ( b )

a(Node)에서 b(Node) 방향으로 '수강하다' 라는 Relationship 생성

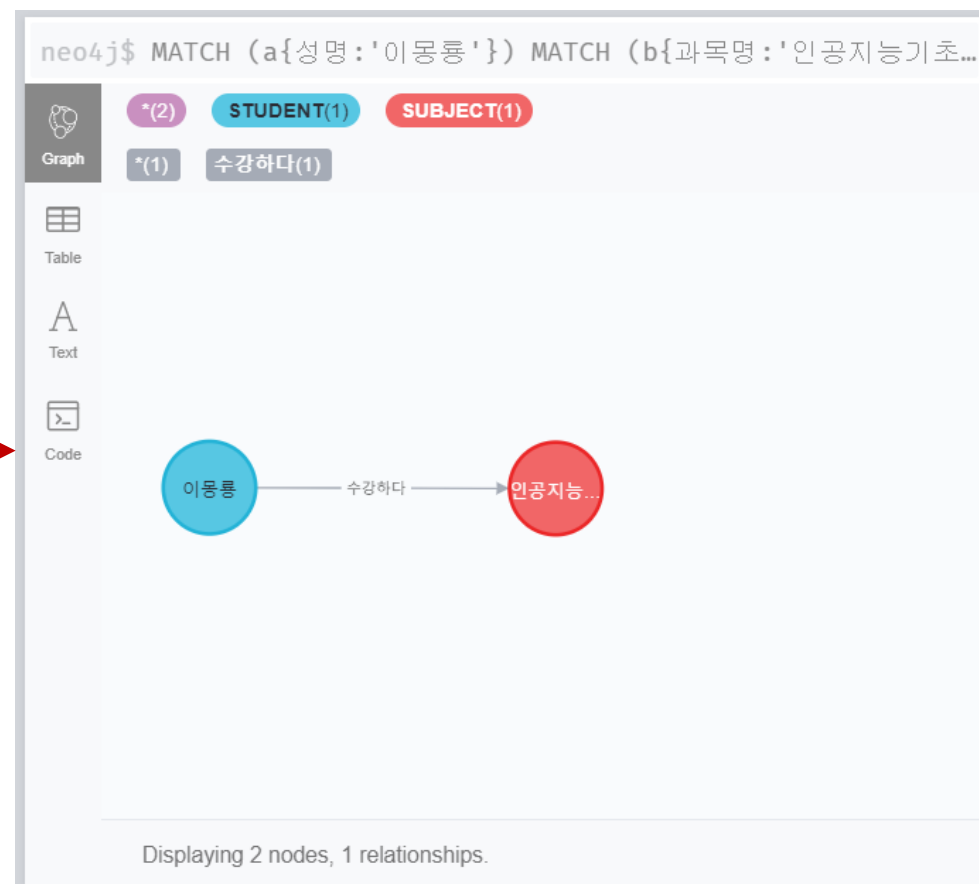
RETURN a, b, r

a, b, r를 리턴하여 출력

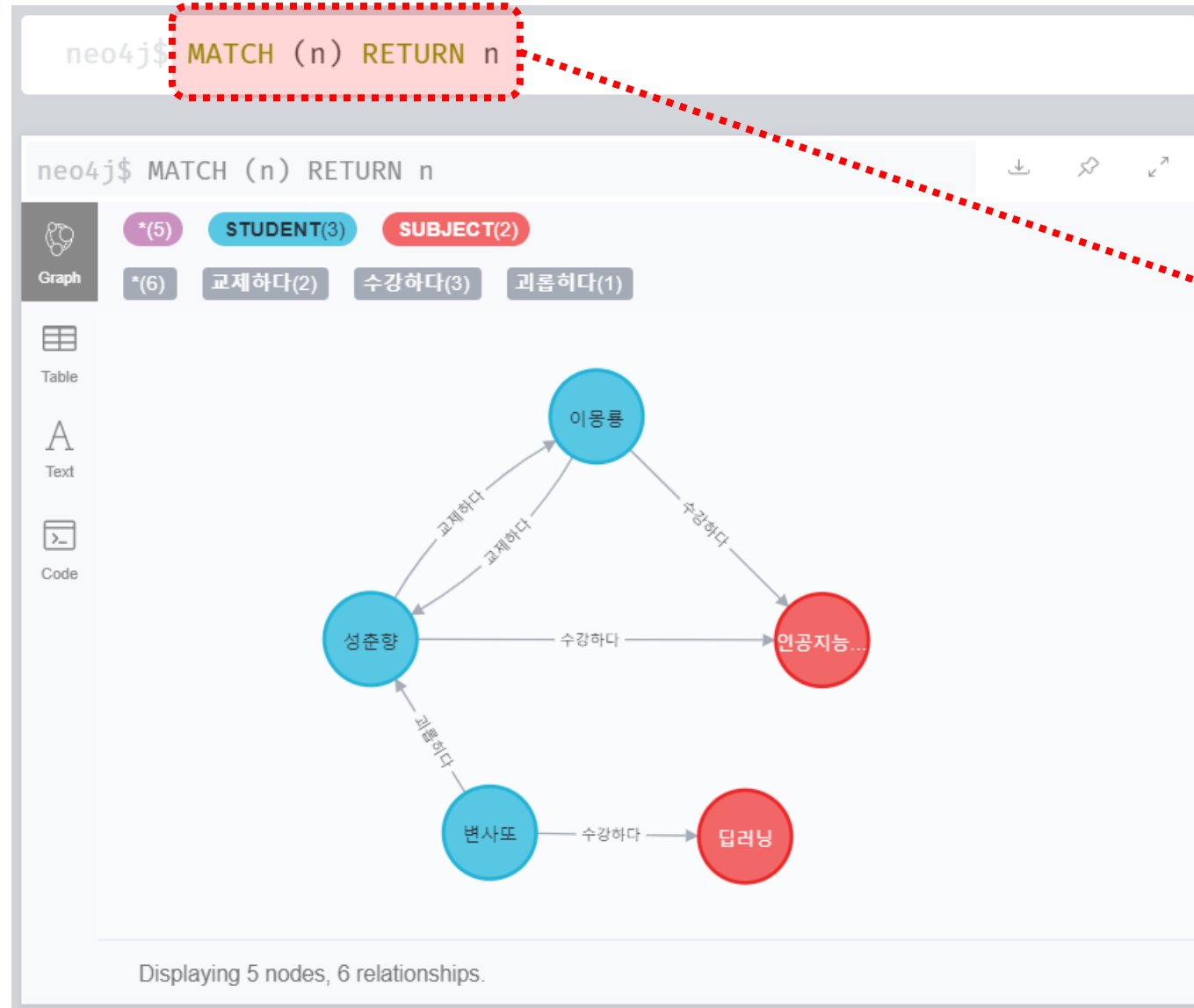


### ▪ Cypher Query CREATE - Relationship 생성

```
1 MATCH (a{성명:'이몽룡'}) MATCH (b{과목명:'인공지능기초'})  
2 CREATE (a) -[r:수강하다] → (b)  
3 RETURN a, b, r
```



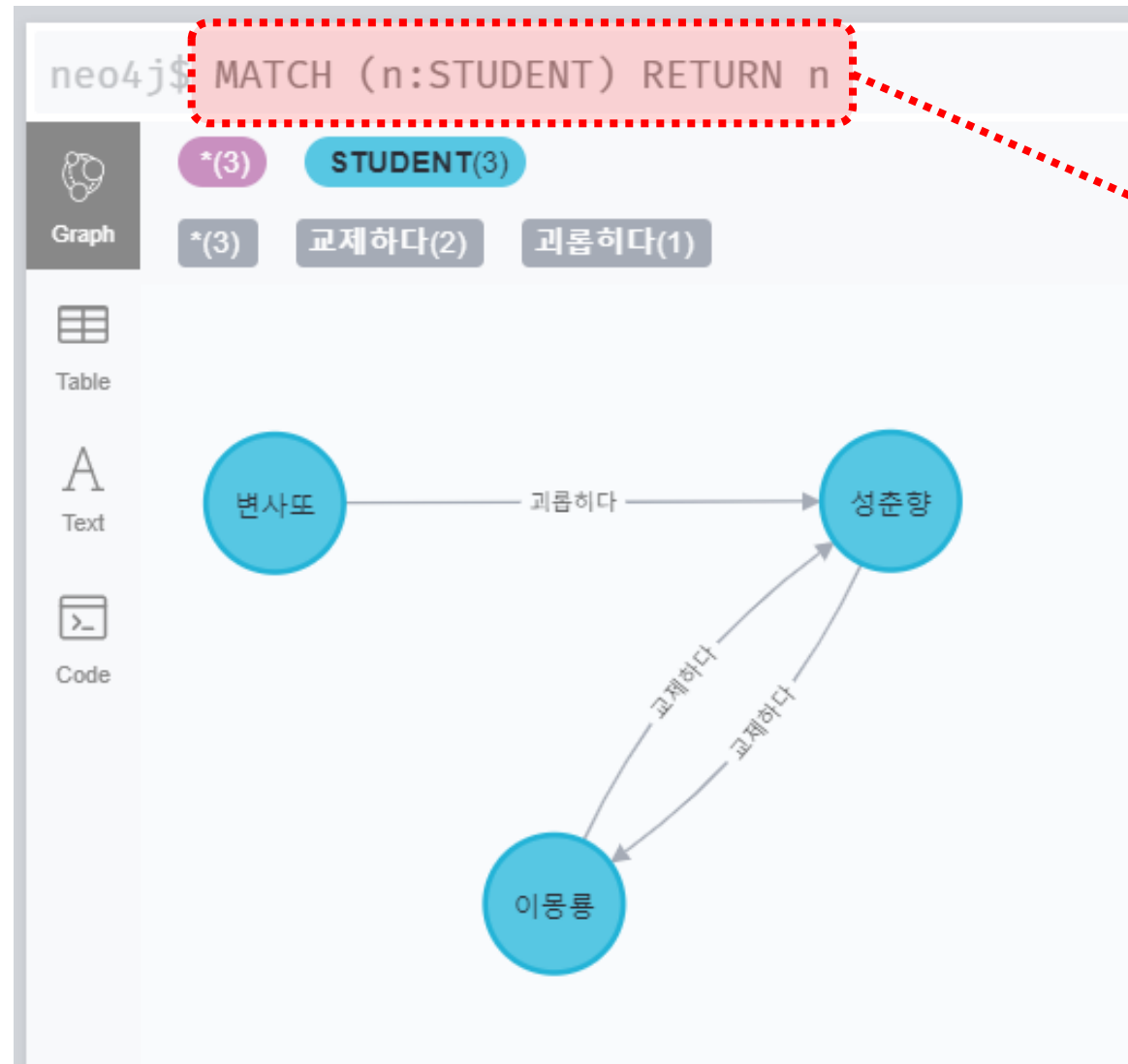
▪ Cypher Query CREATE - 생성 결과 조회



저장된 모든 Node  
리턴 및 출력

# Neo4j Node, Relationship 생성 실습

### ▪ Cypher Query MATCH



STUDENT Label 모두 출력

### ▪ Cypher Query MATCH

The image shows the Neo4j web interface. At the top, the query editor contains the Cypher query: `MATCH (n:SUBJECT { 학점 : 3 }) RETURN n`. This query is highlighted with a red dashed box. Below the query editor, the left sidebar contains icons for Graph, Table, Text, and Code. The 'Table' icon is highlighted with a red dashed box. A red dotted arrow points from the 'Table' icon to the JSON result. Another red dotted arrow points from the query editor to the text '학점이 3학점인 STUDENT Label 모두 출력'. The JSON result is displayed in the main area, showing a single record for a node with identity 4, labels ['SUBJECT'], and properties including '과목명': '딥러닝', '담당 교수': '박 교수', and '학점': 3. At the bottom, a status message reads: 'Started streaming 1 records after 1 ms and completed after 3 ms.'

```
neo4j> MATCH (n:SUBJECT { 학점 : 3 }) RETURN n
```

n

```
{
  "identity": 4,
  "labels": [
    "SUBJECT"
  ],
  "properties": {
    "과목명": "딥러닝",
    "담당 교수": "박 교수",
    "학점": 3
  }
}
```

Started streaming 1 records after 1 ms and completed after 3 ms.

학점이 3학점인  
STUDENT Label 모두 출력

### ▪ Cypher Query MATCH

The image shows the Neo4j Cypher Query Editor interface. The query entered is: `MATCH (a:STUDENT)-[:수강하다]→(b:SUBJECT{과목명:'딥러닝'}) RETURN a.성명`. The results are displayed in a table with one column, `a.성명`, and one row containing the value `"변사또"`. A red dashed box highlights the query, and a red dashed arrow points from it to the explanatory text on the right.

neo4j\$ MATCH (a:STUDENT)-[:수강하다]→(b:SUBJECT{과목명:'딥러닝'}) RETURN a.성명

a.성명
"변사또"

Started streaming 1 records after 1 ms and completed after 3 ms.

딥러닝 SUBJECT와 수강하다  
관계를 가지는 STUDENT의  
성명을 출력  
딥러닝을 수강하는 학생의  
성명을 출력

### ▪ Cypher Query MATCH

The image shows the Neo4j Cypher Query Interface. At the top, the query is entered: `MATCH (a:STUDENT) WHERE a.학년 >= '3학년' RETURN a.성명`. The query is highlighted with a red dashed border. Below the query, the results are displayed in a table view. The table has a header row with the column name `a.성명`. There are two data rows: the first row has the value `"이몽룡"` and the second row has the value `"성준향"`. The table view is selected in the left sidebar. At the bottom, a status message reads: "Started streaming 2 records after 1 ms and completed after 2 ms."

	a.성명
1	"이몽룡"
2	"성준향"

Started streaming 2 records after 1 ms and completed after 2 ms.

STUDENT Label의  
Node 중에서 학년 속성이  
'3학년' 이상인 학생의  
성명 출력

### ▪ Cypher Query MATCH

The image shows the Neo4j Cypher Query Editor interface. The query entered is: `MATCH (a:STUDENT)-[:수강하다]→(b:SUBJECT{과목명:'딤편닝'}) RETURN a.성명`. The results are displayed in a table with one column, `a.성명`, and one row containing the value `"변사또"`. A red dashed box highlights the query, and a red dashed arrow points from it to the explanatory text on the right.

neo4j\$ MATCH (a:STUDENT)-[:수강하다]→(b:SUBJECT{과목명:'딤편닝'}) RETURN a.성명

a.성명
"변사또"

Started streaming 1 records after 1 ms and completed after 3 ms.

딤편닝 SUBJECT와 수강하다  
관계를 가지는 STUDENT의  
성명을 출력  
딤편닝을 수강하는 학생의  
성명을 출력



## Cypher Query MATCH

The image shows the Neo4j Cypher Query Editor interface. The query entered is: `MATCH (a:STUDENT)-[:교제하다]→(b:STUDENT{성명:'성춘향'}) MATCH (a) - [:수강하다] → (d:SUBJECT) RETURN d.과목명`. The query is split into two parts by a semicolon. The first part is highlighted with a red dashed box, and the second part is highlighted with a purple dashed box. The results are displayed in a table with one column, `d.과목명`, and one row containing the value `"인공지능기초"`. A red dotted arrow points from the first part of the query to the text: `성춘향과 교제하는 STUDENT Label의 Node를 'a' 변수 저장`. A purple dotted arrow points from the second part of the query to the text: `a변수에 저장된 Node가 수강하는 SUBJECT Label의 과목명 출력`. At the bottom, a status message reads: `Started streaming 1 records after 2 ms and completed after 4 ms.`

```
neo4j$ MATCH (a:STUDENT)-[:교제하다]→(b:STUDENT{성명:'성춘향'}) MATCH (a) - [:수강하다] → (d:SUBJECT) RETURN d.과목명
```

d.과목명
"인공지능기초"

성춘향과 교제하는  
STUDENT Label의  
Node를 'a' 변수 저장

a변수에 저장된 Node가  
수강하는 SUBJECT Label의  
과목명 출력

Started streaming 1 records after 2 ms and completed after 4 ms.

# Neo4j Cypher Query MATCH 실습

추가 학습 : <https://neo4j.com/docs/cypher-manual/4.1/>