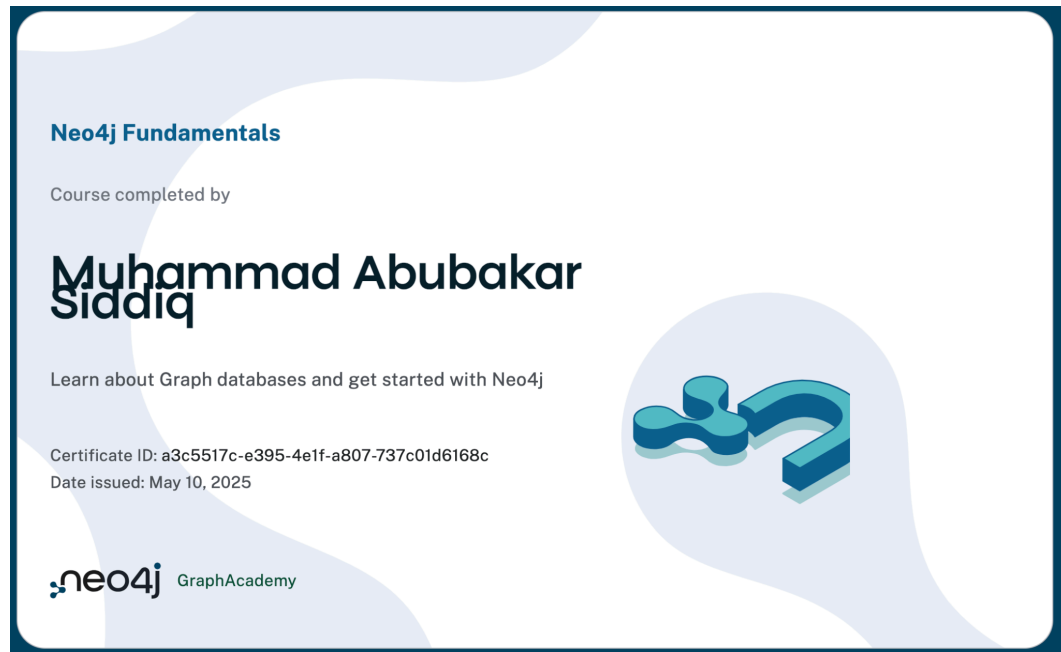


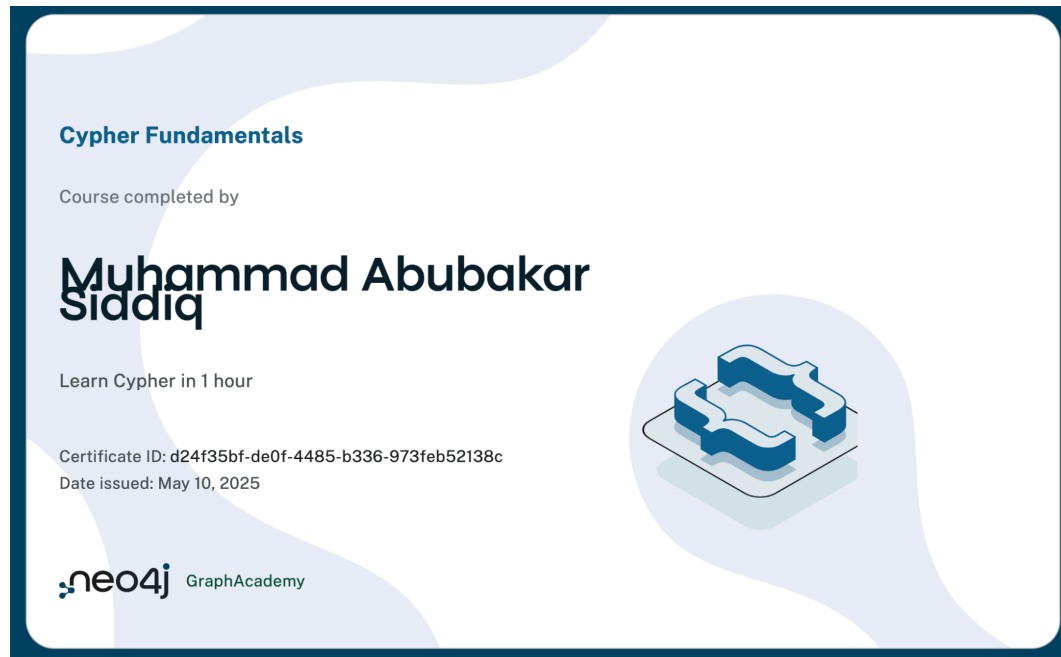
LAMPIRAN



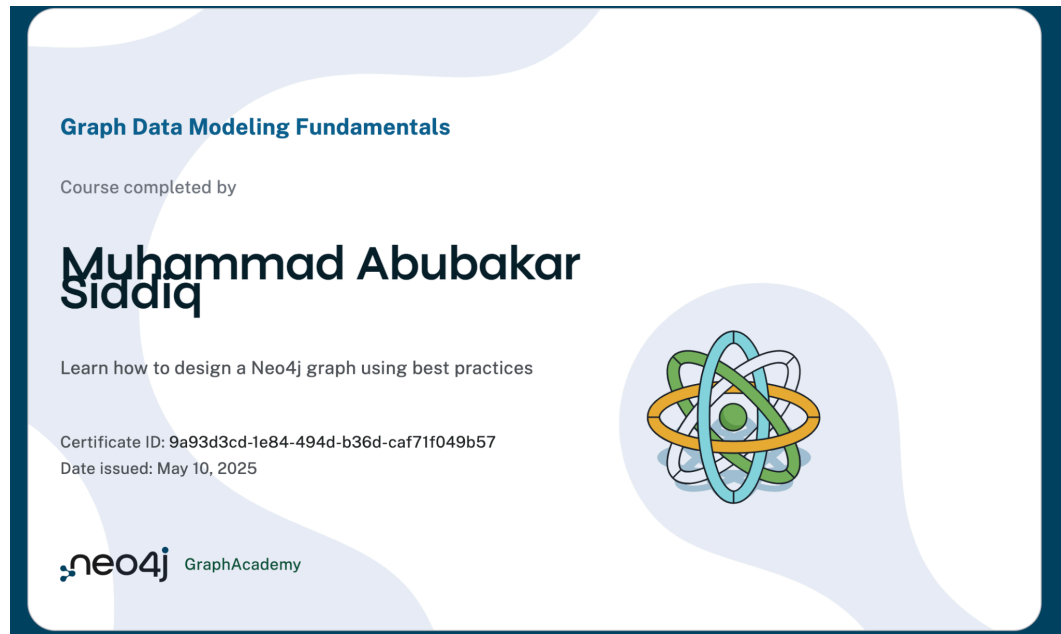
Sertifikat Kelulusan Neo4j Certified Professional



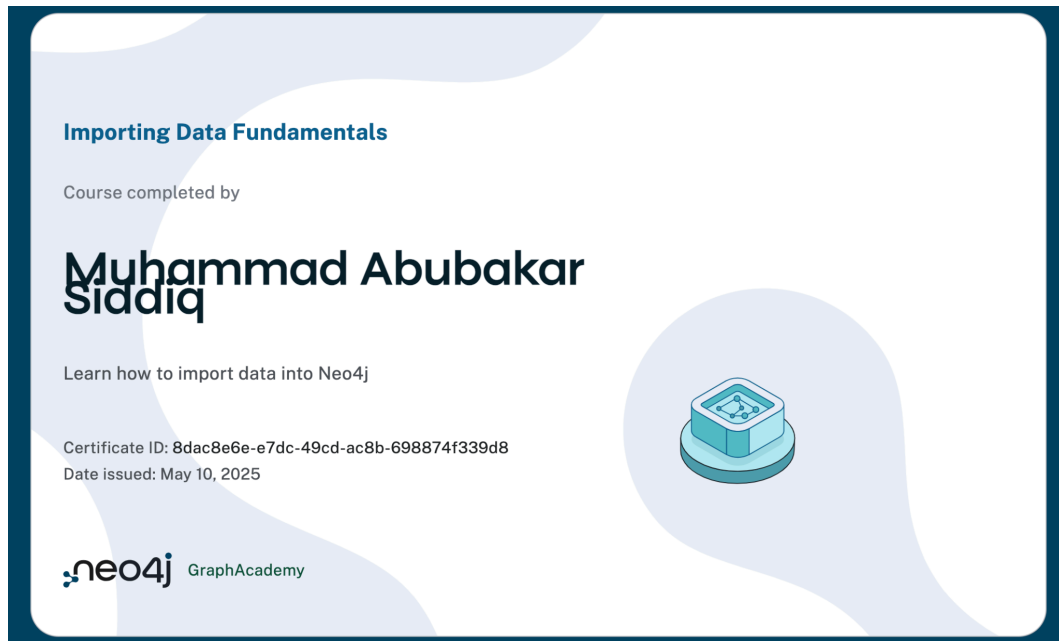
Sertifikat Kelulusan *Neo4j Fundamentals*



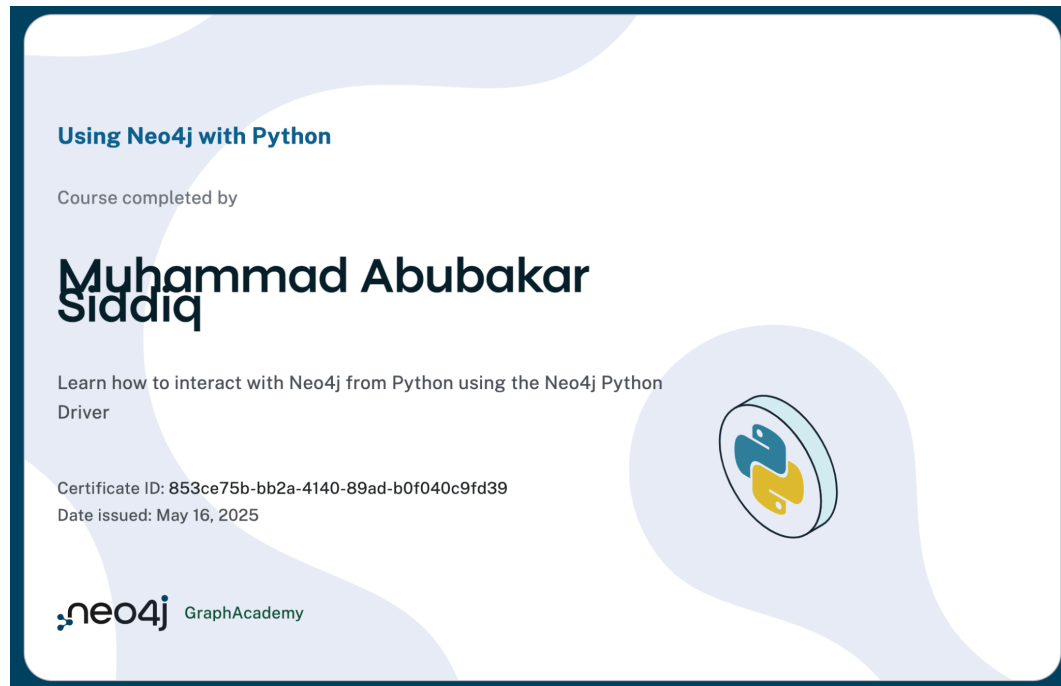
Sertifikat Kelulusan *Cypher Fundamentals*



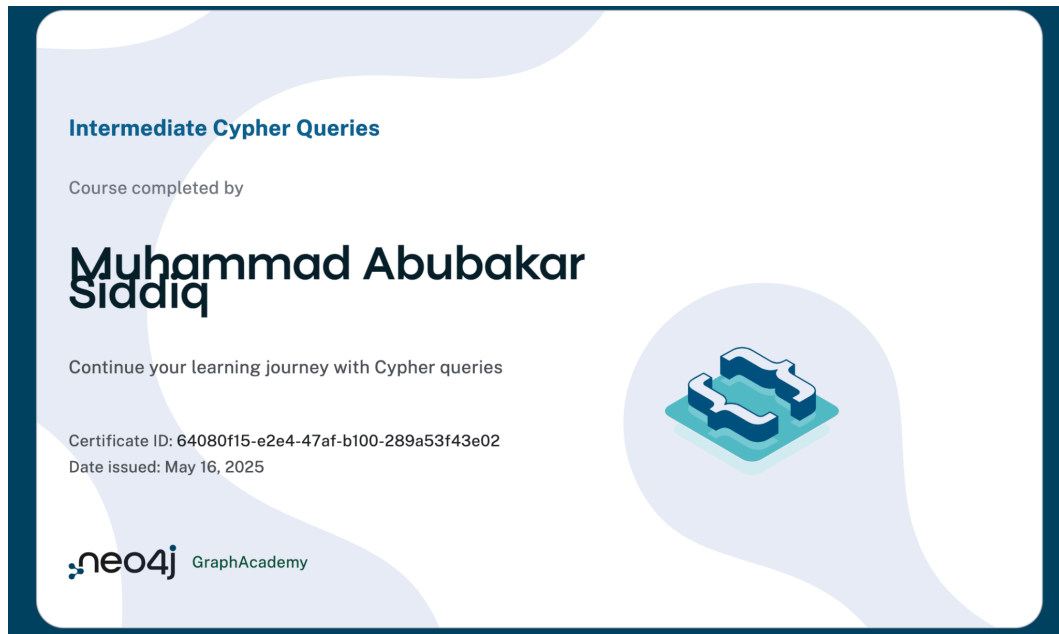
Sertifikat Kelulusan *Graph Data Modeling Fundamentals*



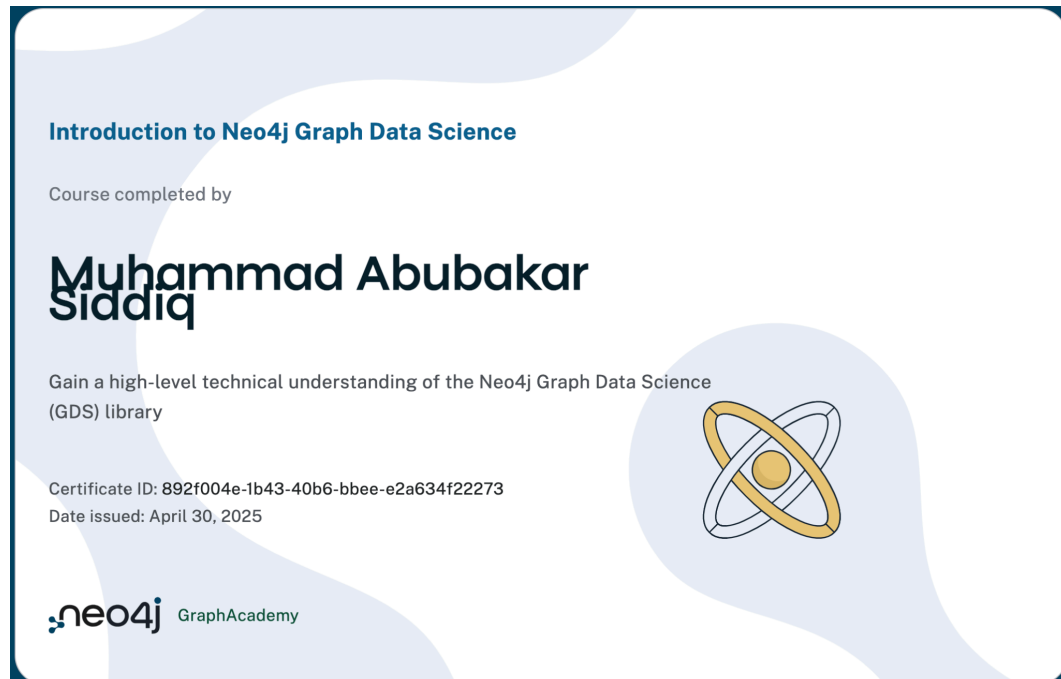
Sertifikat Kelulusan *Importing Data Fundamentals*



Sertifikat Kelulusan *Using Neo4j With Python*



Sertifikat Kelulusan *Intermediate Cypher Queries*



Sertifikat Kelulusan *Introduction to Neo4j Graph Data Science*

actors are the most influential in terms of the number of other actors they have been in recent, high-grossing movies with.

For the sake of this example, we will call a movie "recent" if it was released on or after 1990, and high-grossing if it had revenue >= \$1M.

The graph is not set up to answer this question well with a direct native projection. However, we can use a cypher projection to filter to the appropriate nodes and perform an aggregation to create an `ACTED_WITH` relationship that has a `actedWithCount` property going directly between actor nodes.

The Cypher query to create this data set would be:

```
cypher
MATCH (source:Actor)-[:ACTED_IN]->(m:Movie)-[:ACTED_IN]-(target)
WHERE m.year >= 1990 AND m.revenue >= 1000000
RETURN source.name, count(*) AS actedWithCount
```

The actor's name and the count of movies they acted in with other actors in recent, high-grossing movies are returned.

You can use this query to create a Cypher projection.

```
cypher
MATCH (source:Actor)-[:ACTED_IN]->(m:Movie)-[:ACTED_IN]-(target)
WHERE m.year >= 1990 AND m.revenue >= 1000000
WITH source, target, count(*) AS actedWithCount
WITH gds_graph.project(
  cypher-proj = {
    source,
    target,
    { relationshipProperties: {
      actedWithCount: actedWithCount
    } } AS g
) AS g
RETURN
g.graphname AS graph, g.nodeCount AS nodes, g.relationshipCount AS
```

Results of the first query:

graph	nodes	rels
"cypher-proj"	1887	42648

Results of the second query:

source.name	actedWithCount
"John Heart"	12
"Mancusky Cukky"	10
"Carter Stern"	9
"Joe Frost"	30
"Patric Gueyze"	15
"Whong Goldberg"	24

Dokumentasi Pelatihan *Introduction to Neo4j Graph Data Science*

Video Introduction to Cypher

Watch Read

Diagram illustrating the Cypher query language structure:

```
graph LR
    P1[Person] --> M[Movie]
    P2[Person] --> M
    P3[Person] --> M
    P4[Person] --> M
    P5[Person] --> M
    M --> P6[Person]
```

Check your understanding

1. Read data

Which Cypher clause do you use to read data from Neo4j?

☐ FIND

☐ SELECT

Dokumentasi Pelatihan *Cypher Fundamentals*

The node labels for the graph include:

- Person
- Actor
- Director
- Movie
- Game
- User

The relationships for the graph include:

- ACTED_IN (with an optional role property)
- DIRECTED (with an optional role property)
- RATED (with rating and timestamp properties)
- IN_GENRE

Also notice that the nodes have a number of properties, along with the type of data that will be used for each property. It is important that you understand the property types defined in the data model.

You can view the property types for nodes in the graph by executing this query:

```
cypher
CALL db.schema.nodeTypeProperties()
```

You can view the property types for relationships in the graph by executing this query:

```
cypher
CALL db.schema.relTypeProperties()
```

Each node with a given label has a property that uniquely identifies the node. These nodes are indexed in the graph.

- Movie nodes use movieId
- Person nodes use imdbId

nodeType	nodeLabels	propertyNames	propertyTypes	mandatory
"Game"	["Game"]	"name"	["String"]	true
"User"	["User"]	"name"	["String"]	true
"User"	["User"]	"username"	["String"]	false
"Director/Person"	["Director", "Person"]	"name"	["String"]	true
"Director/Person"	["Director", "Person"]	"movieId"	["String"]	false
"Director/Person"	["Director", "Person"]	"username"	["String"]	true

Started showing 47 records after 178 ms and consumed after 1162 ms.

Overview

Node labels

- Game
- User
- Director
- Person
- Movie

Relationship types

- ACTED_IN
- DIRECTED
- RATED
- IN_GENRE

Displaying 4 nodes & relationships

Dokumentasi Pelatihan *Intermediate Cypher Queries*

Welcome back, Muhammad Abubakar Siddiqi

Congratulations on completing Neo4j Certified Professional!

Here is a link to your certificate. Why not share your achievement on LinkedIn?

<https://graphacademy.neo4j.com/c/8be6a29f-651a-47d2-91b5-eed8d077a7ba>

Beginner

You have completed 57% of this learning path.

[View Learning Path](#)

- Cypher Essentials
- Data Modeling Fundamentals
- Graph Data Modeling Essentials
- Neo4j Fundamentals
- Cypher Fundamentals
- Graph Data Modeling Fundamentals
- Importing Data Fundamentals

Your path to certification

Pass the Neo4j Certified Professional exam with a score of 80% or higher to earn an exclusive badge.

We recommend taking the following courses before attempting the certification.

- Neo4j Fundamentals
- Cypher Fundamentals
- Graph Data Modeling Fundamentals
- Importing Data Fundamentals
- Intermediate Cypher Queries
- Using Neo4j with Python

Congratulations!

You passed Neo4j Certified Professional May 18, 2025

[View your certificate](#)

[Continue where you left off](#)

Dokumentasi Profil Peserta Pelatihan dan Silabus Pelatihan