

```
= Few-shot Examples
:order: 3
:type: lesson
```

Even though you have provided an LLM with specific instructions, it can still make mistakes.

In this lesson, you will learn about Few-Shot examples and how to use them to improve the performance of the LLM.

link:<https://www.promptingguide.ai/techniques/fewshot>[Few-Shot Prompting^] is a technique where you provide the LLM with an example of how to respond or generate a response to a specific scenario.

== Example Cypher

Here is a use case demonstrating a problem that would benefit from Few-Shot prompting.

You ask the LLM to generate a Cypher statement to answer the question:

What movies has Tom Hanks directed and what are the genres?

The LLM generates the following Cypher statement:

```
MATCH (p:Person)-[:DIRECTED]->(m:Movie) WHERE p.name = 'Tom Hanks'
RETURN m.genres
```

The Cypher statement returns the following result:

```
[{'genres': null}, {'genres': null}]
```

From this data, the LLM can understand that Tom Hanks has directed two movies, but there is no information about the genres of those movies.

The generated Cypher statement is wrong because it uses the `m.genres` property, which doesn't exist. Instead, it should follow the `:IN_GENRE` relationship to `:Genre` nodes and use the `.name` property.

You can improve the LLM by providing an example of a correct Cypher statement in the Cypher generation prompt:

Examples:

```
Find movies and genres:
MATCH (m:Movie)-[:IN_GENRE]->(g)
RETURN m.title, g.name
```

[%collapsible]

.Click to reveal the full Cypher generation prompt

====

[source,python]

```
include::code/cypher-gen-few-shot.py[tag=template]
```

```
----
```

```
=====
```

The LLM can use the example to help it generate the correct Cypher statement:

```
MATCH (p:Person)-[:DIRECTED]->(m:Movie)-[:IN_GENRE]->(g:Genre)
WHERE p.name = 'Tom Hanks'
RETURN DISTINCT g.name
```

Which returns the correct data:

```
[{'g.name': 'Drama'}, {'g.name': 'Comedy'}, {'g.name': 'Romance'}]
```

[NOTE]

```
=====
```

The few-shot example is not the complete Cypher statement to answer the question, but it is enough to show the LLM how to use the `[:IN_GENRE]` relationship.

```
=====
```

The LLM can now also generate the correct Cypher statement for other questions involving movies and genres:

What genre of film is Toy Story?

```
MATCH (m:Movie {title: 'Toy Story'})-[:IN_GENRE]->(g:Genre)
RETURN g.name
```

```
[{'g.name': 'Adventure'}, {'g.name': 'Animation'}, {'g.name':
'Children'}, {'g.name': 'Comedy'}, {'g.name': 'Fantasy'}]
```

You can provide multiple examples for different scenarios to return more accurate results.

For example, finding roles actors played in movies:

Find roles for actors:

```
MATCH (m:Movie)-[r:ACTED_IN]->(p:Person)
WHERE m.title = 'movie title' AND p.name = 'actor name'
RETURN m.title, r.role, p.name
```

Few-shot prompts allow you to provide targeted examples to the LLM to improve its performance.

Providing examples can improve the LLM's performance on a specific task or the performance in general.

== Check Your Understanding

```
include::questions/1-few-shot-prompts.adoc[leveloffset=+1]
```

[.summary]

== Summary

In this lesson, you learned about Few-Shot prompting and how to use it to improve the performance of the LLM.

In the next optional challenge, you will add the Cypher generation chain to an agent and give it conversation memory.