

```
= Providing Specific Instructions
:order: 2
:type: lesson
```

In this lesson, you will learn how to provide instructions to the LLM to improve the responses to questions.

You provide instructions and context to the LLM in the prompt.

```
== Following the schema
```

In the previous lesson, you used this prompt to generate Cypher statements:

```
[source,python]
----
include:../1-cypher-qa-chain/code/cypher-gen.py[tag=template]
----
```

The LLM's training data included many Cypher statements, but these statements were not specific to the structure of your graph database. As a result, the LLM may generate Cypher statements that are not valid and do not conform to the schema.

You can provide specific instructions to the LLM to state that the generated Cypher statements should follow the schema.

For example, you could give instructions only to use the provided relationship types and properties in the schema.

```
[source,python]
----
include::code/cypher-gen-follow-schema.py[tag=template]
----
```

```
[TIP]
```

```
.Instructions not rules
```

There is no right or wrong way to provide instructions to the LLM. The instructions are unlike traditional programming tasks where you would supply rules or definitions.

You may have to experiment to find the right balance between providing too much or too little instruction.

For example, the instructions are expressed as both a positive and negative statement. You will do this. You won't do this.

```
    Use only the provided relationship types and properties in the schema.
    Do not use any other relationship types or properties that are not
provided.
```

The results of this change may not be immediately apparent when you run the program. Instructions allows you to fine-tune the LLM's responses and make them more consistent.

== Understanding data

The LLM may also need additional instructions about the data. For example, how data is formatted or how to handle missing data.

If you ask the LLM to generate Cypher for the question:

Who acted in The Matrix and what roles did they play?

The LLM will generate the correct Cypher:

```
MATCH (m:Movie {title: 'The Matrix'})<-[:ACTED_IN]-(a:Actor)
RETURN a.name as Actor, a.role as Role
```

However, the graph database returns no data because movies that start with `The` are renamed as `{title}, The`. `The Matrix` is stored as `Matrix, The`.

You can provide instructions to the LLM to fix this problem.

For movie titles that begin with "The", move "the" to the end, For example "The 39 Steps" becomes "39 Steps, The" or "The Matrix" becomes "Matrix, The".

[source, python]

```
----
include::code/cypher-gen-understand-data.py[tag=template]
----
```

If you ask the LLM the same question, it should now format the title correctly and return the correct data:

```
MATCH (m:Movie {title: 'Matrix, The'})<-[:ACTED_IN]-(a:Actor)
RETURN a.name as Actor, a.role as Role
```

== Controlling the response

As well as instructing the LLM on how to deal with the question, you can also instruct the LLM on how to respond.

You could instruct the LLM only to respond when the Cypher statement returns data.

If no data is returned, do not attempt to answer the question.

You may want the LLM to only respond to questions in the scope of the task. For example:

Only respond to questions that require you to construct a Cypher statement.

Do not respond to any questions that might ask anything else than for

you to construct a Cypher statement.

Concise responses from the LLM may be needed:

Do not include any explanations or apologies in your responses.

Or you may want to restrict the format of the response:

Do not include any text except the generated Cypher statement.

Ultimately, you must fine-tune your instructions for the specific task to ensure the best results.

== Check Your Understanding

```
include::questions/1-instructions.adoc[leveloffset=+1]
```

```
[.summary]
```

```
== Summary
```

In this lesson, you learned something how to provide the LLM with instructions to improve the responses to questions.

In the next lesson, you will learn about few-shot examples and how they can improve the LLM's performance.