1. Create a Node

CREATE (:Person {name: "Alice", age: 30})

```

-Explanation: This command creates a single node labeled `Person` with two properties: `name` set to "Alice" and `age` set to 30. Nodes represent entities in the graph database.

2. View Created Nodes

MATCH (p:Person) RETURN p

- Explanation: This command retrieves all nodes with the label `Person`. The `MATCH` clause searches for nodes, and `RETURN p` specifies that we want to see the matched nodes. It allows you to verify the nodes you've created in the database.

3. Create a Relationship

MATCH (a:Person {name: "Alice"}), (b:Person {name: "Bob"})

CREATE (a)-[:FRIEND\_OF]->(b)

```

- Explanation: This command establishes a relationship between two existing `Person` nodes ("Alice" and "Bob"). The `MATCH` clause finds the nodes, and the `CREATE` clause defines a `FRIEND\_OF` relationship from Alice to Bob. This relationship allows you to model connections between entities.

4. Delete a Node

MATCH (p:Person {name: "Alice"}) DETACH DELETE p

```

- Explanation: This command deletes the node labeled `Person` with the name "Alice". The `DETACH` keyword is used to remove any relationships connected to this node before deletion, ensuring that no orphaned relationships remain. This is important to maintain data integrity in the graph.

New commands

Here are some completely different commands with unique names that have not been mentioned before. These commands will allow you to create a graph, visualize the data, and explore relationships.

### 1. \*\*Create a Unique Social Network\*\*

```cypher

CREATE

(s:Person {name: "Oscar", age: 24}),

(t:Person {name: "Paula", age: 29}),

(u:Person {name: "Quentin", age: 32}),

(v:Person {name: "Rachel", age: 27}),

(w:Person {name: "Sam", age: 30}),

(s)-[:COLLEAGUE\_OF]->(t),

(t)-[:FRIEND\_OF]->(u),

(u)-[:FRIEND\_OF]->(v),

(v)-[:COLLEAGUE\_OF]->(w),

(w)-[:FRIEND\_OF]->(s)

```

- \*\*Explanation\*\*: This creates a network of five `Person` nodes (Oscar, Paula, Quentin, Rachel, and Sam) with `COLLEAGUE\_OF` and `FRIEND\_OF` relationships among them.

### 2. \*\*Create a Book and Author Graph\*\*

```cypher

CREATE

(book1:Book {title: "Journey to the Center of the Earth", genre: "Science Fiction"}),

(book2:Book {title: "Pride and Prejudice", genre: "Romance"}),

(author1:Author {name: "Jules Verne"}),

(author2:Author {name: "Jane Austen"}),

(author1)-[:WROTE]->(book1),

(author2)-[:WROTE]->(book2)

```

- \*\*Explanation\*\*: This creates two `Book` nodes (Journey to the Center of the Earth and Pride and Prejudice) and two `Author` nodes (Jules Verne and Jane Austen), establishing a `WROTE` relationship.

### 3. \*\*Create a Movie and Actor Graph\*\*

```cypher

CREATE

(movie1:Movie {title: "Inception", year: 2010}),

(movie2:Movie {title: "The Matrix", year: 1999}),

(actor1:Actor {name: "Leonardo DiCaprio"}),

(actor2:Actor {name: "Keanu Reeves"}),

(actor1)-[:STARS\_IN]->(movie1),

(actor2)-[:STARS\_IN]->(movie2)

```

- \*\*Explanation\*\*: This creates two `Movie` nodes (Inception and The Matrix) and two `Actor` nodes (Leonardo DiCaprio and Keanu Reeves), establishing a `STARS\_IN` relationship.

### 4. \*\*Visualize All Nodes and Relationships\*\*

To visualize all nodes and their relationships, you can run:

```cypher

MATCH (n)-[r]->(m)

RETURN n, r, m

```

- \*\*Explanation\*\*: This retrieves all nodes (`n`) and their outgoing relationships (`r`) to other nodes (`m`), allowing you to visualize the entire graph structure.

### 5. \*\*Find All Relationships of a Specific Person\*\*

For example, find all relationships involving Oscar:

```cypher

MATCH (o:Person {name: "Oscar"})-[r]->(friends)

RETURN o, r, friends

```

- \*\*Explanation\*\*: This retrieves all relationships that Oscar has with other nodes, allowing you to see his connections in the graph.

### 6. \*\*Count the Number of Books by Each Author\*\*

```cypher

MATCH (a:Author)-[:WROTE]->(b:Book)

RETURN a.name AS AuthorName, COUNT(b) AS NumberOfBooks

```

- \*\*Explanation\*\*: This counts how many books each author has written and returns the results.

### 7. \*\*Delete a Node and Its Relationships\*\*

To delete a node (for example, Paula), you can use:

```cypher

MATCH (p:Person {name: "Paula"})

DETACH DELETE p

```

- \*\*Explanation\*\*: This command deletes the node for Paula along with any relationships she has, ensuring no orphaned relationships remain.

Feel free to run these commands in your Neo4j environment. They are designed to help you visualize different kinds of relationships and structures in your graph. Let me know if you need more examples or further assistance! Good luck with your practical exam!