

0. Database Generation

The screenshot shows the Neo4j Aura web interface. On the left is a sidebar with navigation options: Get started, Developer hub, Data services (Instances, Import, Graph Analytics, Data APIs, Agents), and Tools (Query). The main panel displays 'Database information' for an instance named 'HomeworkDB'. It shows 153 nodes and 385 relationships. The nodes are categorized by labels: Cusine, Member, Owner, and Restaurant. The relationships are categorized by types: OWNS, REVIEWED, and TYPE_OF. A table of property keys is visible, listing attributes like atmosphere, birth, cid, city, email, fname, lname, mid, name, phone, price, rid, service, state, street, taste, username, value, and zip. The right-hand pane shows a successful Cypher query execution: `neo4j$ CREATE (:Member:Owner {mid: 1020, fname: 'Cecil', lname: 'Esparza', username: 'cecesparza', email: 'cecesparza@fak...});` with a completion time of 2.294 ms.

1. Adding Dabo Sweeny to the DB

This screenshot shows the Neo4j Aura interface after adding a new member. The 'Database information' section now shows 154 nodes. The Cypher query editor on the right contains two queries. The first is a match query: `neo4j$ MATCH (m:Member {member_id: 7500}) RETURN m;` which returns a single record for a member with ID 7500. The second query is a create statement: `neo4j$ CREATE (:Member { member_id: 7500, name: "Dabo Swinney", birthYear: 1969, state: "SC", email: "dabo@clmson.edu", username: "dabo"});` which successfully created a new node with 6 properties and 1 label. The completion time for the second query is 28 ms.

2. Setting the Tiger Chow Restaurant

This screenshot shows the Neo4j Aura interface after adding a new restaurant. The 'Database information' section now shows 155 nodes. The Cypher query editor on the right contains two queries. The first is a match query: `neo4j$ MATCH (r:Restaurant {restaurant_id: 10000}) RETURN r;` which returns a single record for a restaurant with ID 10000. The second query is a create statement: `neo4j$ CREATE (:Restaurant { restaurant_id: 10000, name: "Tiger Chow", city: "Clemson", state: "SC"});` which successfully created a new node with 4 properties and 1 label. The completion time for the second query is 21 ms.

3. Updating Tiger Chow

The screenshot shows the Neo4j Aura DB interface. On the left, the 'Database information' panel displays 155 nodes and 385 relationships. The 'Property keys' section lists attributes for various node types. The main query editor shows two queries:

```
neo4j$ MATCH (r:Restaurant {restaurant_id: 10000}) RETURN r;
```

The result shows a single record for the restaurant 'Tiger Chow' with properties: price_rating: 2, city: 'Clemson', phone: '(888) 555-8888', restaurant_id: 10000, name: 'Tiger Chow', state: 'SC'.

```
neo4j$ MATCH (r:Restaurant {restaurant_id: 10000}) SET r.phone = "(888) 555-8888", r.price_rating = 2;
```

The second query successfully updates the phone number and price rating for the restaurant.

4. Creating the Reviewed Relationship

The screenshot shows the Neo4j Aura DB interface. On the left, the 'Database information' panel displays 155 nodes and 386 relationships. The 'Property keys' section lists attributes for various node types. The main query editor shows a query to create a reviewed relationship between a member and a restaurant:

```
neo4j$ MATCH (m:Member {member_id: 7500}), (r:Restaurant {restaurant_id: 10000}) CREATE (m)-[:REVIEWED { taste: 5, service: 5 }]->(r);
```

The result shows a single record for the relationship 'REVIEWED' between the member 'Dabo Summey' and the restaurant 'Tiger Chow'.

The 'Results overview' panel shows the following results:

- Nodes (2): Member (1), Restaurant (1)
- Relationships (1): REVIEWED (1)

The query successfully creates the reviewed relationship between the member and the restaurant.

5. Created Fran's reviewed relationship

I should've asked how you'd wanted these formatted, but I'm pushing both images I figured would be relevant

The screenshot shows the Neo4j Aura web interface. On the left, the 'Database information' panel displays 155 nodes and 387 relationships. The 'Relationships' section lists 'OWNS', 'REVIEWED', and 'TYPE_OF'. The 'Property keys' section lists various attributes like 'atmosphere', 'birth', 'birthYear', 'cid', 'city', 'email', 'fname', 'lname', 'member_id', 'mid', 'name', 'phone', 'price', 'price_rating', 'restaurant_id', 'rid', 'service', 'state', 'street', 'taste', 'username', 'value', and 'zip'. The main query editor shows a Cypher query:

```
neo4j$ MATCH (m:Member {mid: 1089}), (r:Restaurant {restaurant_id: 10000}) CREATE (m)-[:REVIEWED { taste: 4, service: 4, atmo : 2}]->(r);
```

 The results panel shows a single record for the relationship created.

The screenshot shows the Neo4j Aura web interface with the same database information. The main query editor shows a Cypher query:

```
neo4j$ MATCH (m:Member {mid: 1089})-[:REVIEWED]->(r:Restaurant {restaurant_id: 10000}) RETURN m, rev, r;
```

 The results panel shows a single record for the relationship created, with nodes 'Fran' and 'Tiger Chow' connected by a 'REVIEWED' relationship.

6. Queried the restaurants that Fran could stand

neo4j Aura / dwelsh2@clermson.edu / New project

Instance: HomeworkDB Database: neo4j_owners User: Aura (dwelsh2@clermson.edu)

Database information

Nodes (155)

Relationships (388)

Property keys

atmosphere birth birthYear cid city email fname lname member_id mid name phone price price_rating restaurant_id rid service state street taste username value zip

neo4j\$ MATCH (m:Member {mid: 1009})-[rev:REVIEWED]->(r:Restaurant) WHERE rev.value >= 1 RETURN m.fname + ' ' + m.lname AS member, r.name AS restaurant, rev.value AS rating

member	restaurant	value_rating
"Fran Dagwood"	"Butcher Paper Steaks"	2
"Fran Dagwood"	"Marlins"	1
"Fran Dagwood"	"Tiger Chow"	4
"Fran Dagwood"	"Tiger Chow"	4

Started streaming 4 records after 57 ms and completed after 62 ms.

7. Queried the Italian and Seafood Restaurants

neo4j Aura / dwelsh2@clermson.edu / New project

Instance: HomeworkDB Database: neo4j_owners User: Aura (dwelsh2@clermson.edu)

Database information

Nodes (155)

Relationships (388)

Property keys

atmosphere birth birthYear cid city email fname lname member_id mid name phone price price_rating restaurant_id rid service state street taste username value zip

neo4j\$ MATCH (o:Owner)-[:OWNS]->(r:Restaurant)-[:TYPE_OF]->(c:Cuisine) WHERE c.name IN ["Italian", "Seafood"] RETURN c.name AS cuisine, r.name AS restaurant, o.name AS owner

cuisine	restaurant	owner
"Italian"	"Authentica Rustica"	"Dolly Michael"
"Italian"	"The Sicilian"	"Len Sterling"
"Italian"	"Maestro Grill"	"Ashton Johnston"
"Italian"	"The Pasta Bowl"	"Rachel Avery"
"Italian"	"Guldos Cucina"	"Harland Roberts"
"Italian"	"Taste of Rome"	"Ruben Fish"
"Seafood"	"Captain Seafood"	"Carlene Gonzalez"
"Seafood"	"Hook, Line, and Stinker"	"Ezekiel Pineda"
"Seafood"	"Marlins"	"Precious Weir"
"Seafood"	"Boathouse Floaters"	"Leigh Fowler"

Started streaming 13 records after 29 ms and completed after 32 ms.

8. Find the shortest path between Dabo and Herb

neo4j Aura / dwelsh2@clermson.edu / New project

Instance: HomeworkDB Database: neo4j_owners User: Aura (dwelsh2@clermson.edu)

Database information

Nodes (155)

Relationships (388)

Property keys

atmosphere birth birthYear cid city email fname lname member_id mid name phone price price_rating restaurant_id rid service state street taste username value zip

neo4j\$ MATCH (d:Member {mid: 7500}), (h:Member {mid: 1001})

neo4j\$ MATCH (d:Member {fname: "Dabo", lname: "Swinney"}, (h:Member {fname: "Herb", lname: "Christopher"})) MATCH p = shortestPath(d, h)

No changes, no records

Completed after 66 ms