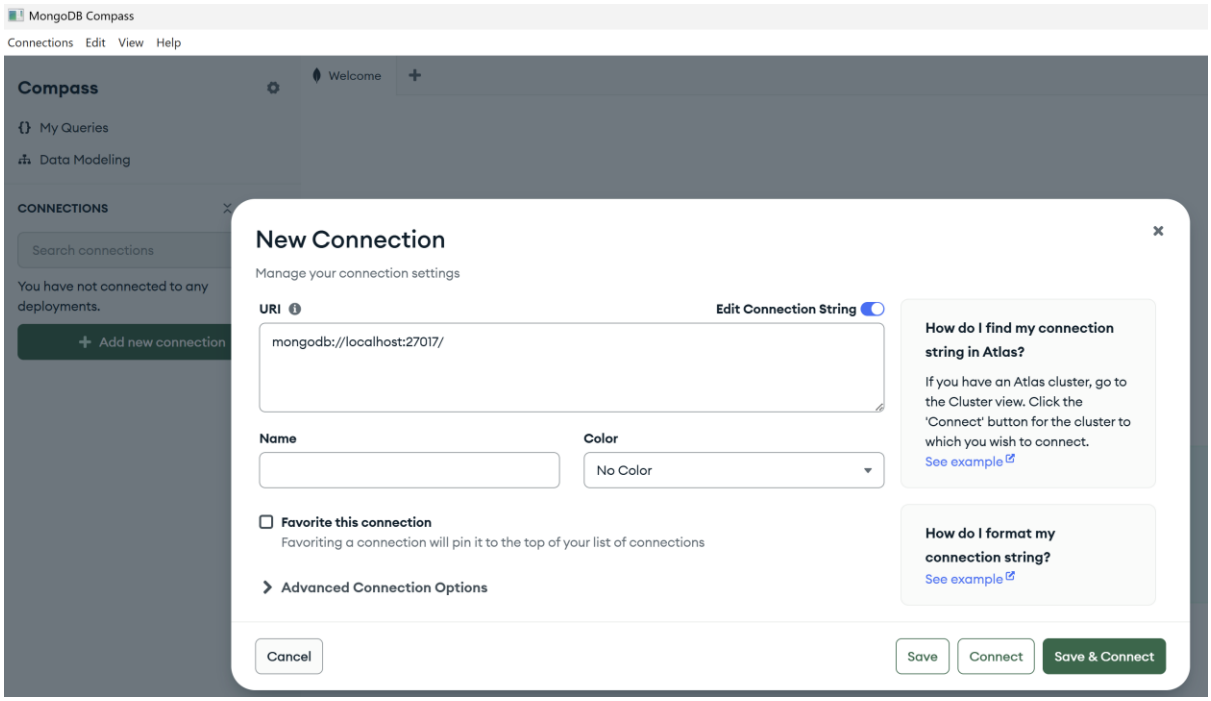
Lab 3

Part 1

First we have to start the mongo database by running the docker command

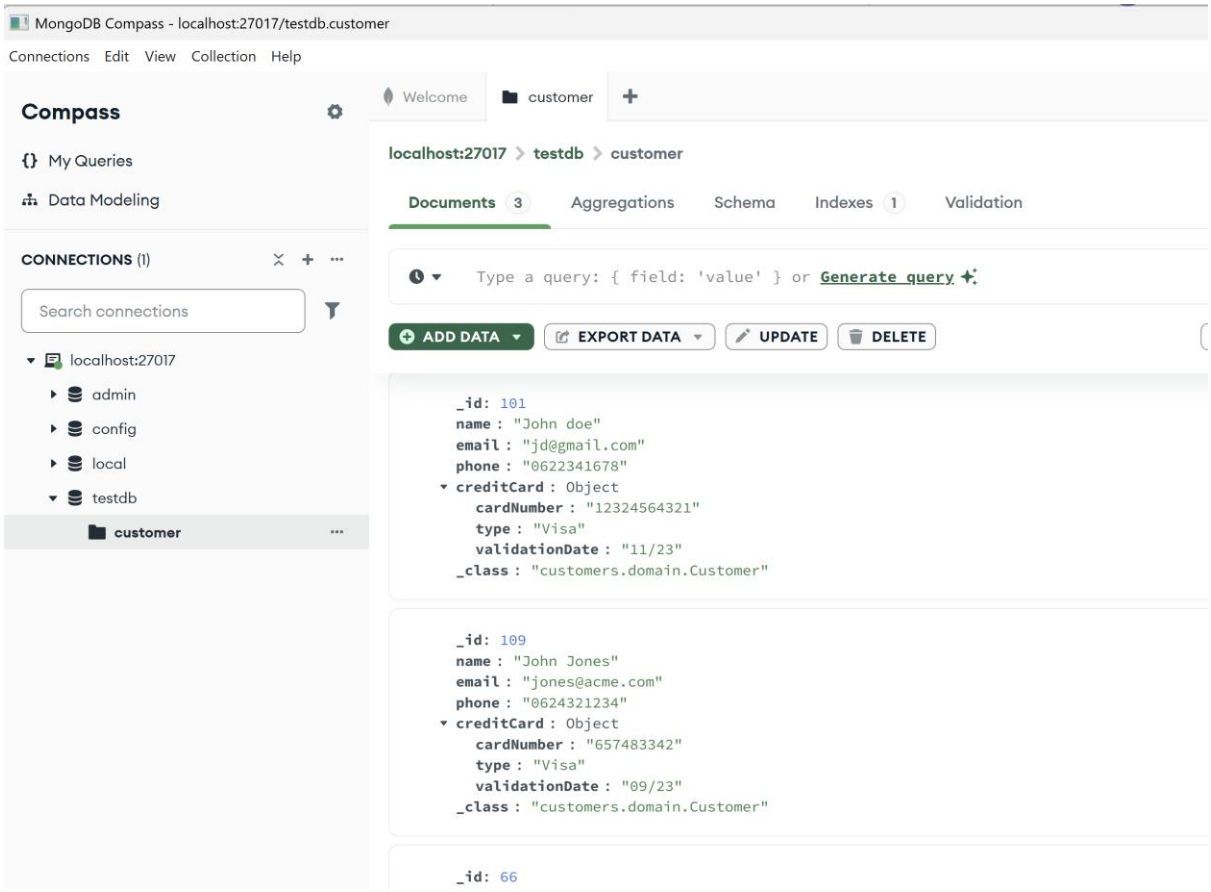
docker run --name mongodb -d -p 27017:27017 mongo:latest

Then download **MongoCompass** from **https://www.mongodb.com/try/download/compass**
Install and start MongoCompass



Click the **Save&Connect** button.

Run the given **Lesson3SpringMongoDemo** application and see that the application saves some customers with creditcards in the **testdb** database.



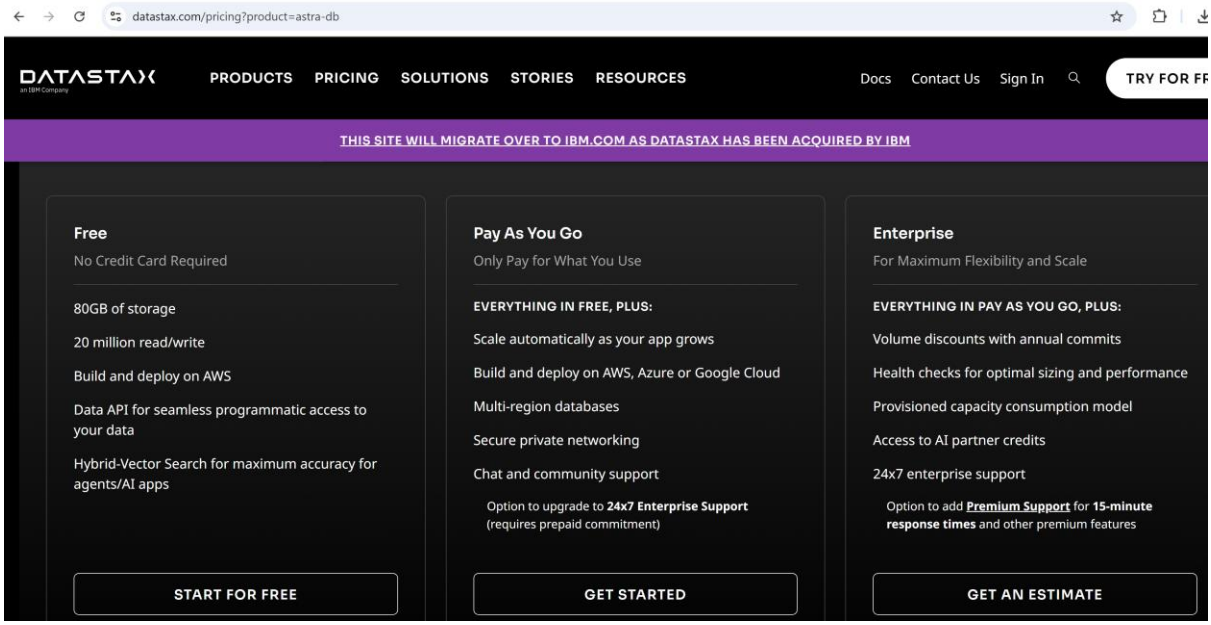
Modify the given **Lesson4SpringMongoDemo** application so that the application stores students in the database
A Student contains the attributes name, phoneNumber and email.
A student has also an address.
Create a class Address with the attributes street, city and zip.
In the application add 5 students in the database.
Then perform the following queries:

- Get all students
- Get all students with a certain name
- Get a student with a certain phoneNumber
- Get all students from a certain city

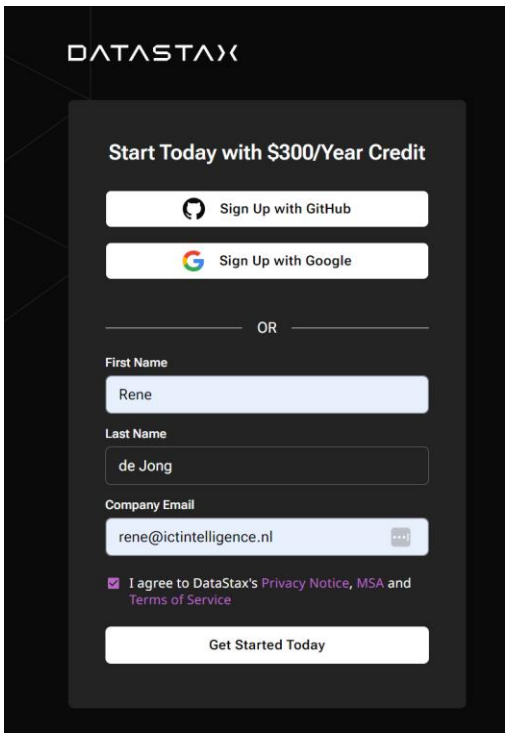
Check the data in the collection using Mongo compass.

Part 2

Go to https://www.datastax.com/pricing/astra-db

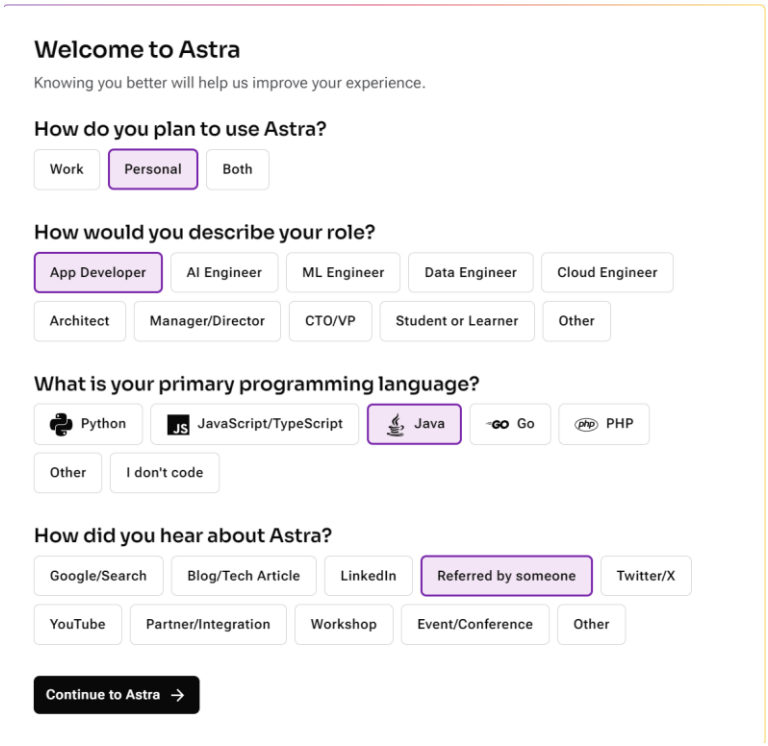


Select the **Free** option

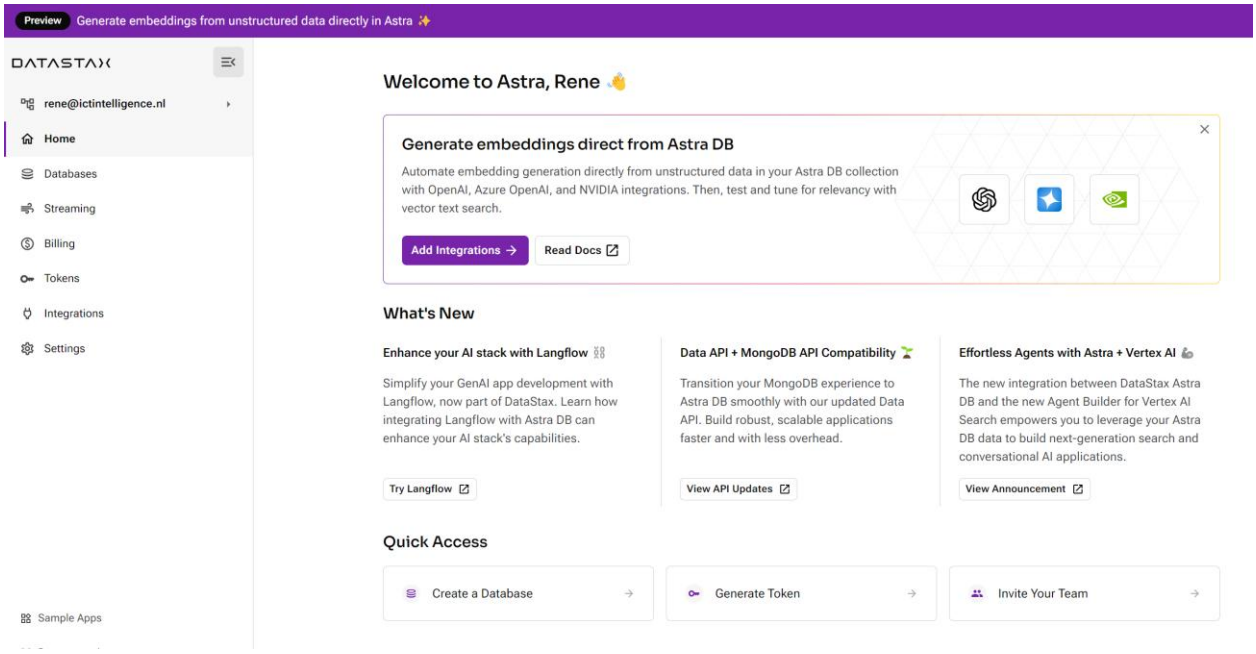


Enter your data and click the **Get Started Today**

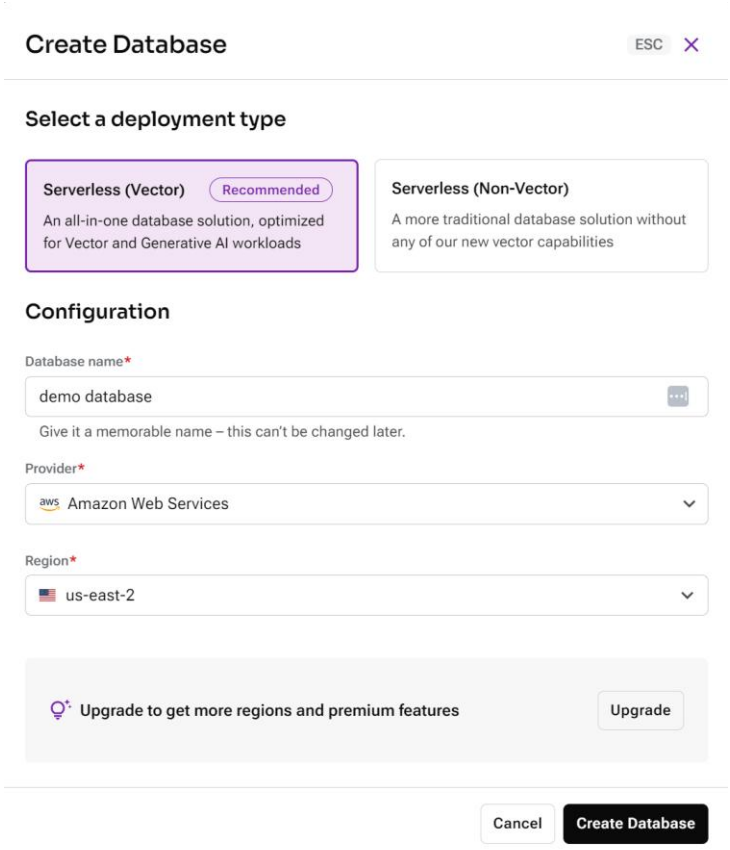
You need to verify your account in your email, and then you need to enter a password.



Click the **Continue to Astra** button.

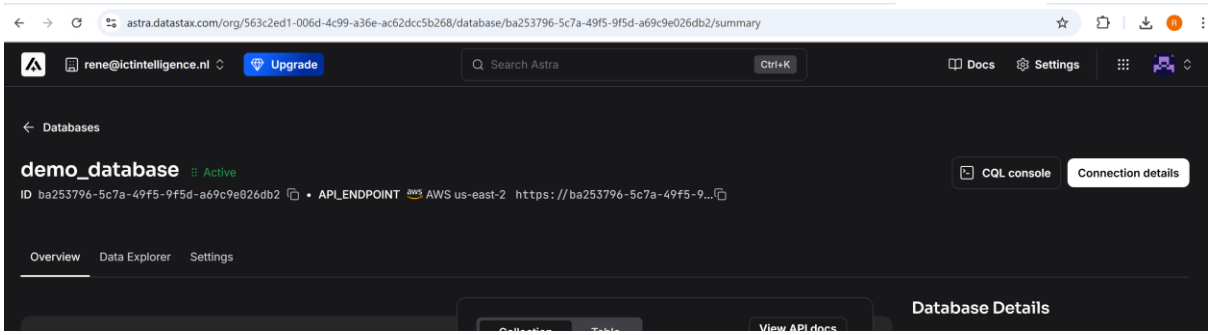


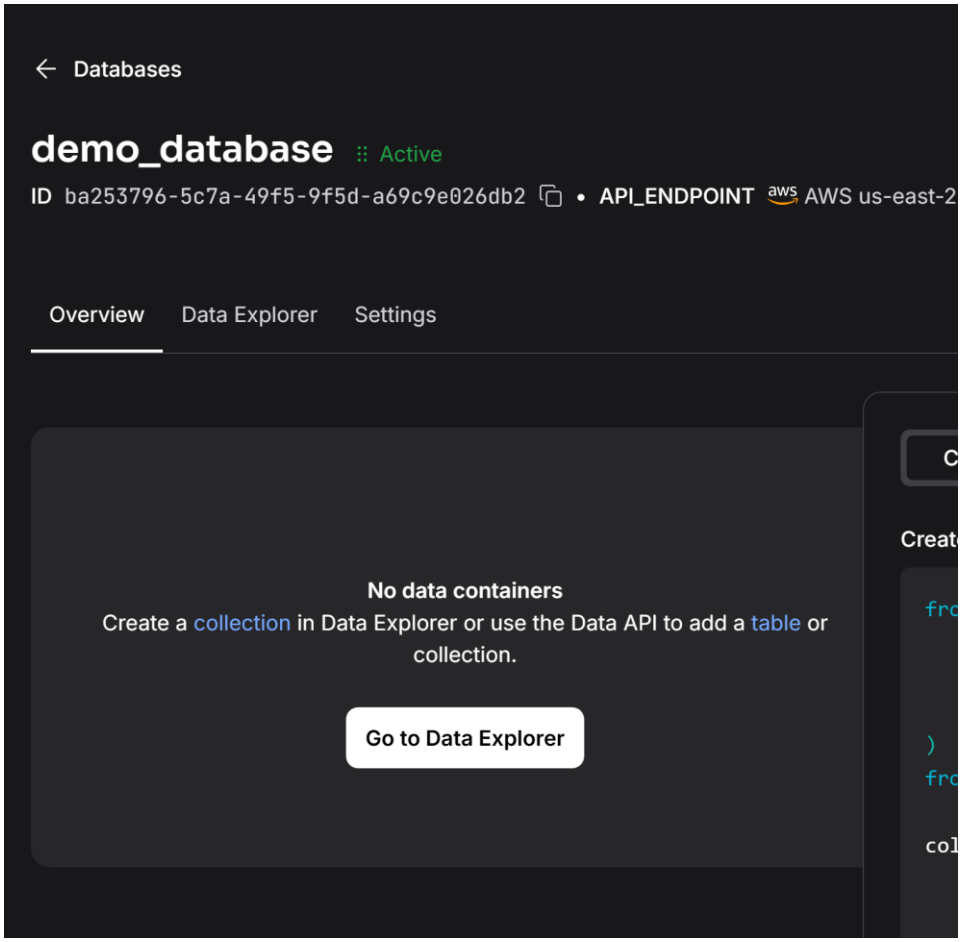
Click **Create a Database**



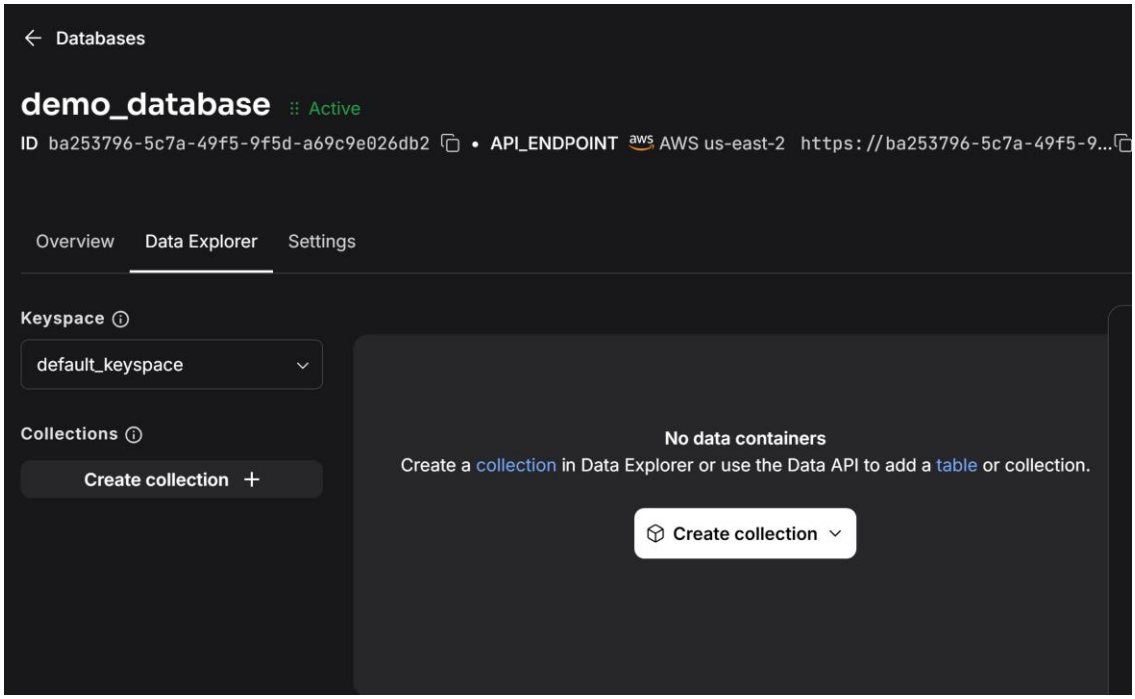
Enter the database name and select a region. Then click **Create Database**

Wait till the database is created

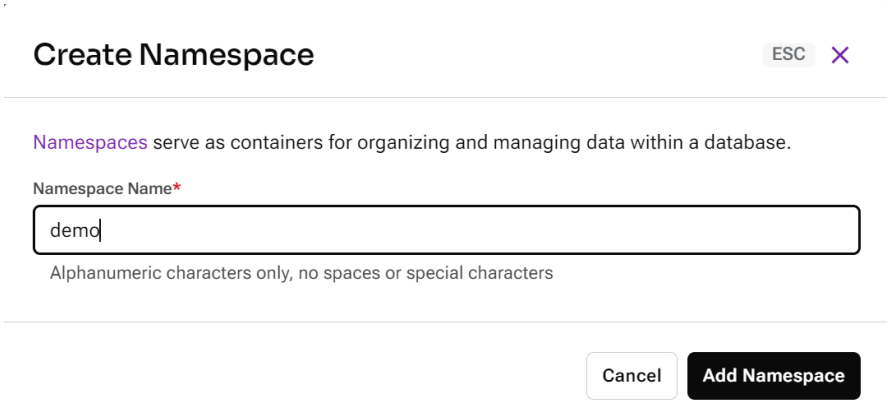




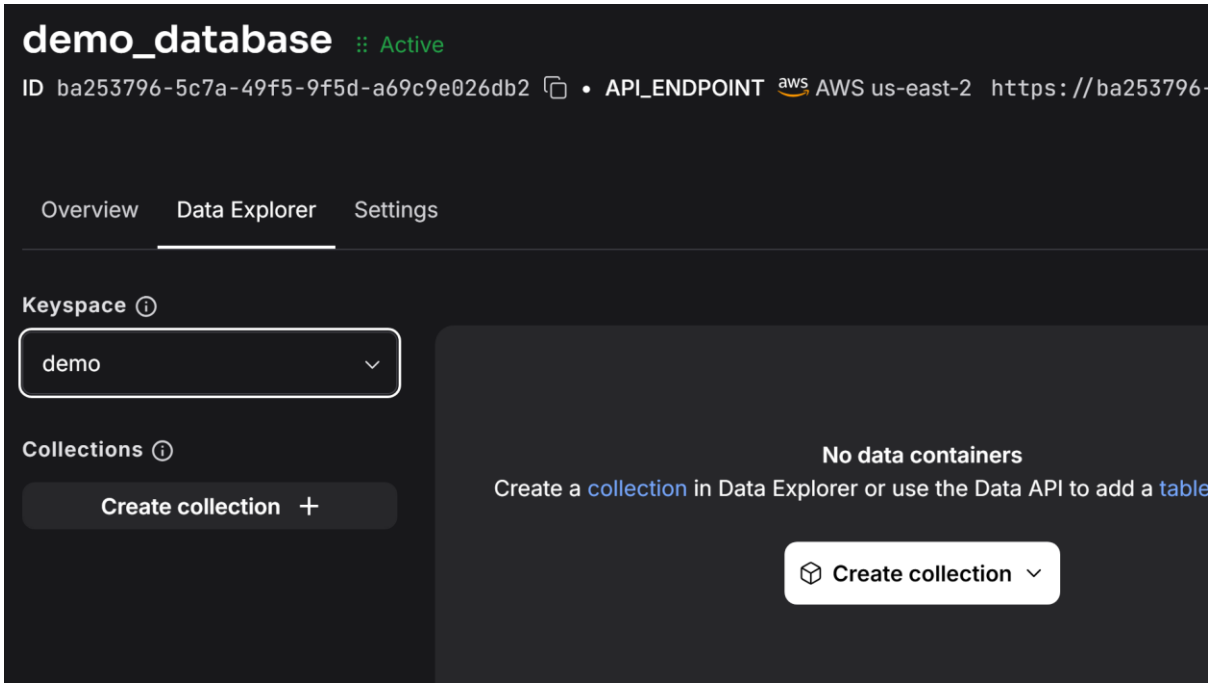
Click the **Go to Data Explorer** button.

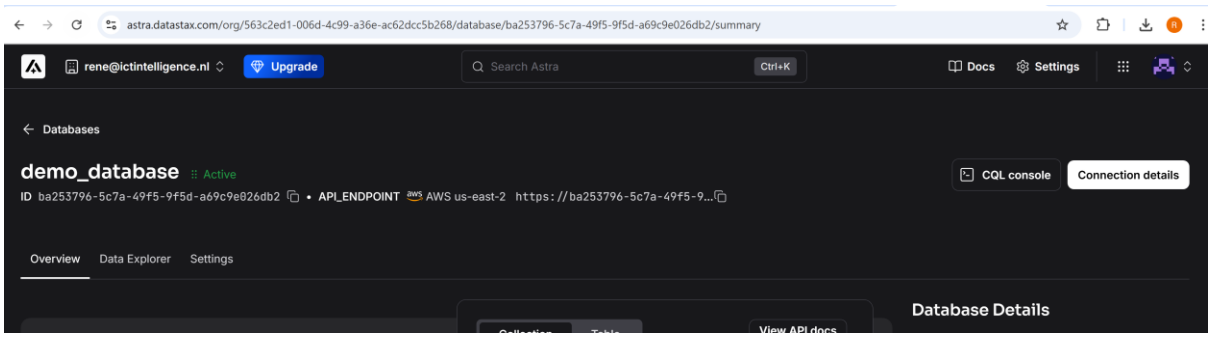


Select the namespace drop down list and select **create a new namespace**

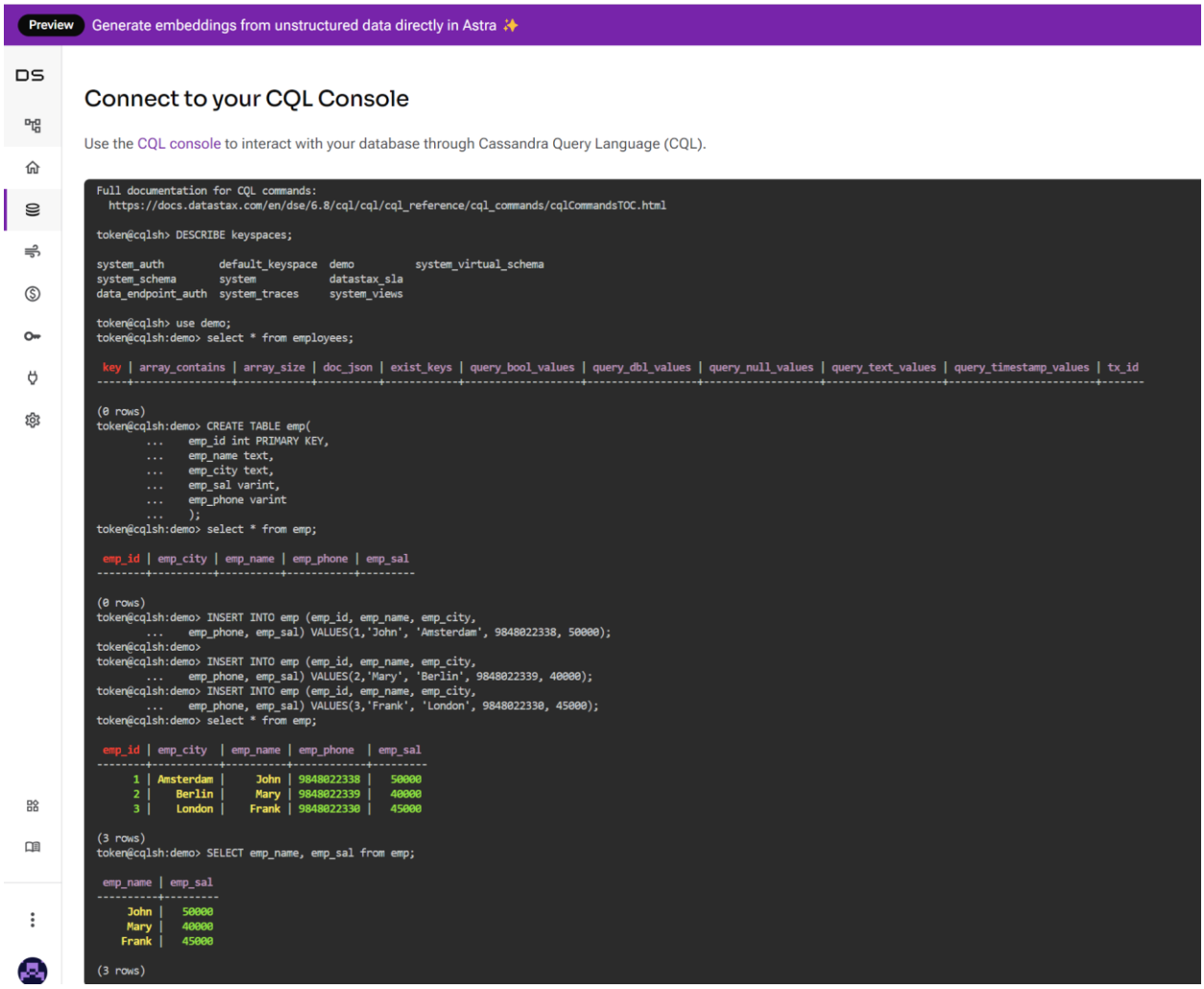


Create the **demo** namespace





Now click the **CQL Console** button



Now write the following commands in the console:

DESCRIBE keyspaces;

USE demo;

CREATE TABLE ratings_by_user (email TEXT, title TEXT, year INT, rating INT, PRIMARY KEY ((email), title, year));

Now insert the following data in the table:

```
INSERT INTO ratings_by_user (email, title, year, rating)
VALUES ('joe@datastax.com', 'Alice in Wonderland', 2010, 9);
INSERT INTO ratings_by_user (email, title, year, rating)
VALUES ('joe@datastax.com', 'Edward Scissorhands', 1990, 10);
INSERT INTO ratings_by_user (email, title, year, rating)
VALUES ('jen@datastax.com', 'Alice in Wonderland', 2010, 10);
INSERT INTO ratings_by_user (email, title, year, rating)
VALUES ('jen@datastax.com', 'Alice in Wonderland', 1951, 8);
```

Perform the following query:

```
SELECT * FROM ratings_by_user;
```

Write the following queries:

Retrieve all data from the user with email 'joe@datastax.com'

```
SELECT * FROM ratings_by_user
WHERE email = 'joe@datastax.com';
```

Retrieve all rating data from the user with email 'joe@datastax.com'for the movie 'Alice in Wonderland' from the year 2010

```
SELECT * FROM ratings_by_user
WHERE email = 'joe@datastax.com'
  AND title = 'Alice in Wonderland'
  AND year = 2010;
```

Now try to get the data for all movies with a rating of 10:

```
SELECT * FROM ratings_by_user
WHERE rating= 10;
```

Notice that this query is not possible because rating is not part of the PK.

Now create the following table:

```
CREATE TABLE ratings_by_movie (
  title TEXT,
  year INT,
  email TEXT,
  rating INT,
  PRIMARY KEY ((title, year), email)
);
```

And populate the table with the following data:

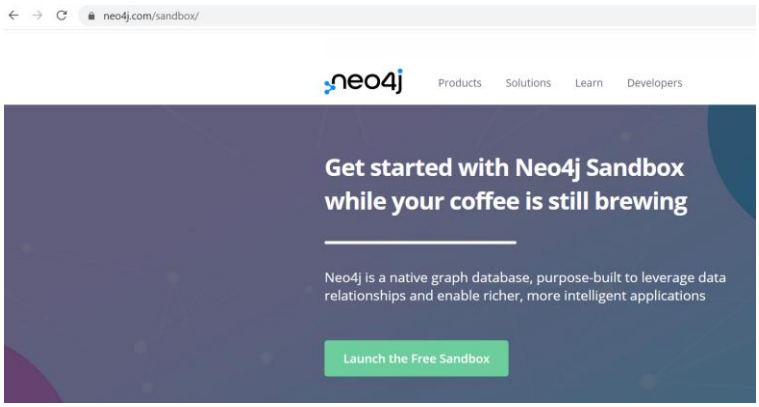
```
INSERT INTO ratings_by_movie (title, year, email, rating)
VALUES ('Alice in Wonderland', 2010, 'jen@datastax.com', 10);
INSERT INTO ratings_by_movie (title, year, email, rating)
VALUES ('Alice in Wonderland', 2010, 'joe@datastax.com', 9);
INSERT INTO ratings_by_movie (title, year, email, rating)
VALUES ('Alice in Wonderland', 1951, 'jen@datastax.com', 8);
INSERT INTO ratings_by_movie (title, year, email, rating)
VALUES ('Edward Scissorhands', 1990, 'joe@datastax.com', 10);
```

Write the query to select all ratings from 'Alice in Wonderland' from 2010

```
SELECT * FROM ratings_by_movie
WHERE title = 'Alice in Wonderland'
  AND year = 2010;
```

Part 3

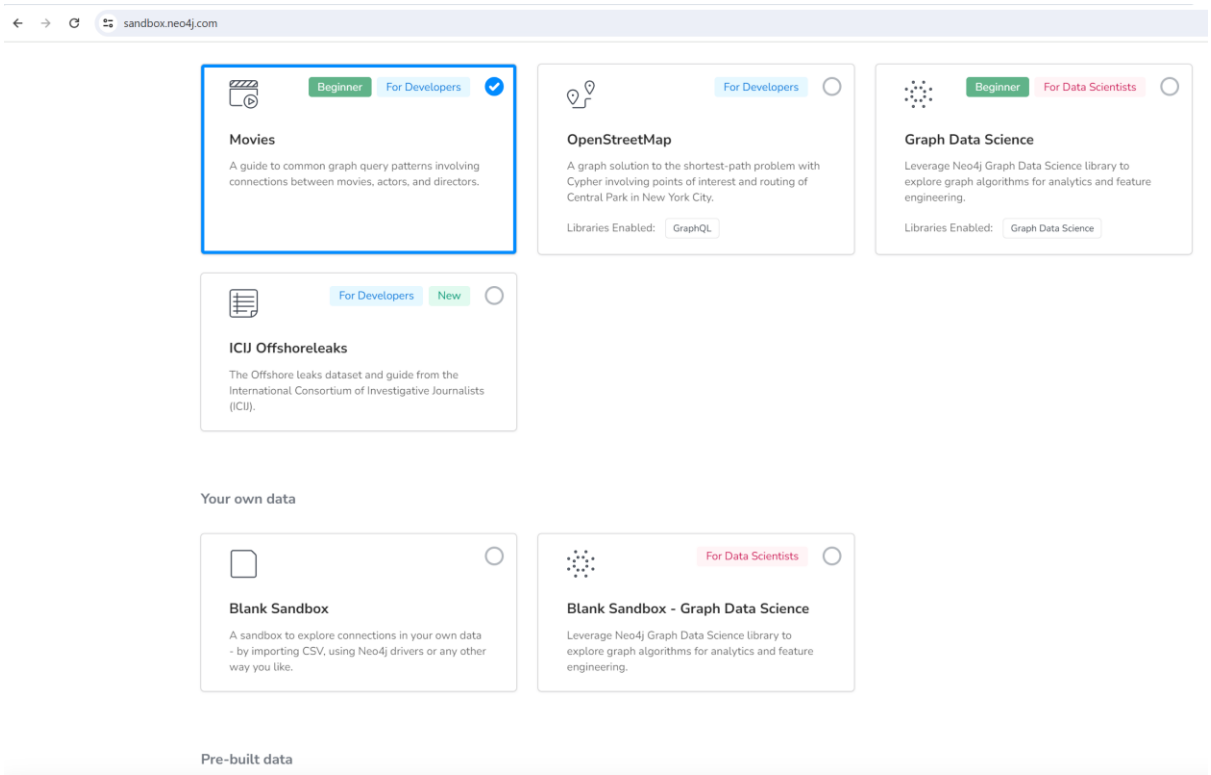
Go to **https://neo4j.com/sandbox/**



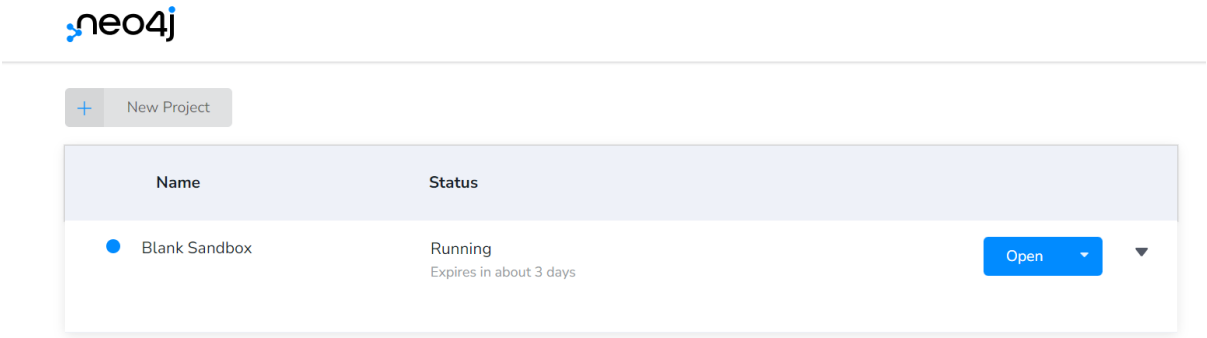
Click the **Launch the free Sandbox** button.

Sign up or login with Google, LinkedIn or Twitter

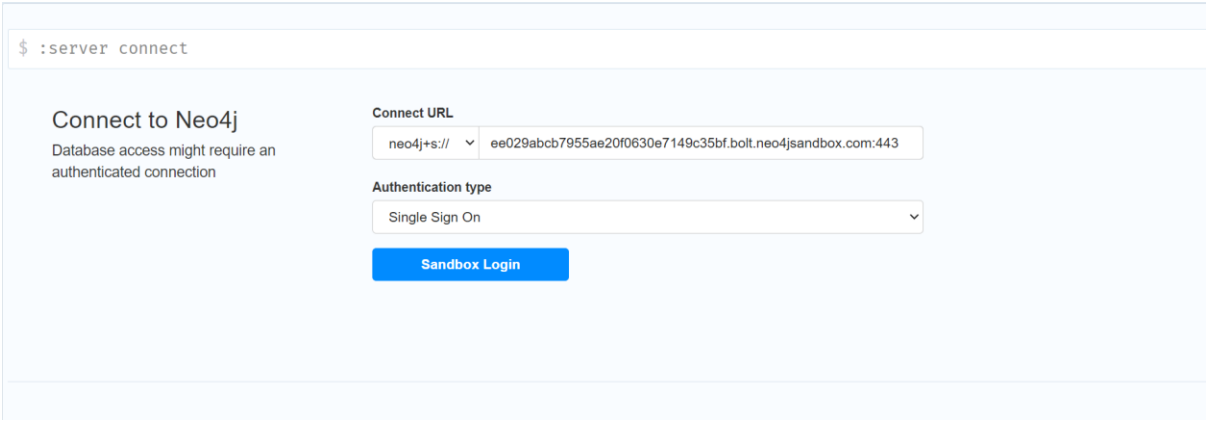
Setup your account and login.



Choose **Blank Sandbox** and **click Create**

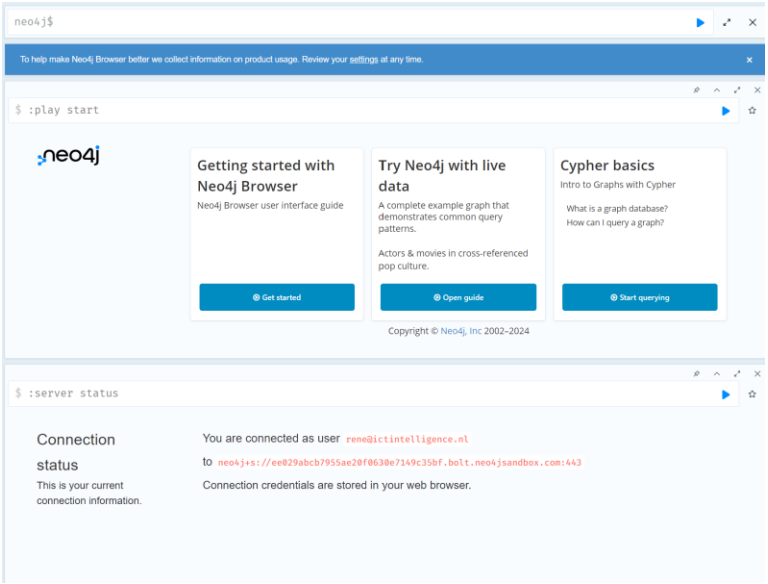


Now **Open** your project.

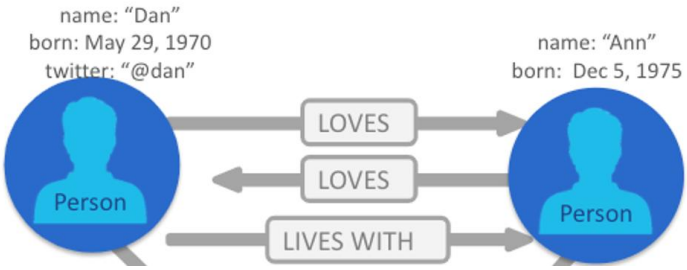


Click **Sandbox Login**

Login with your username and password.



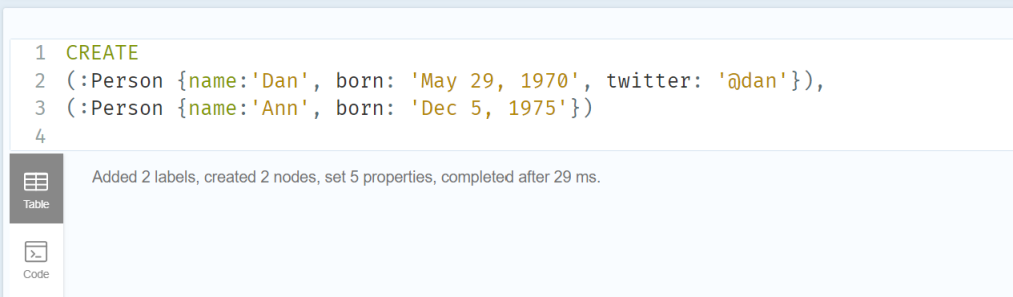
We are going to implement the following graph:



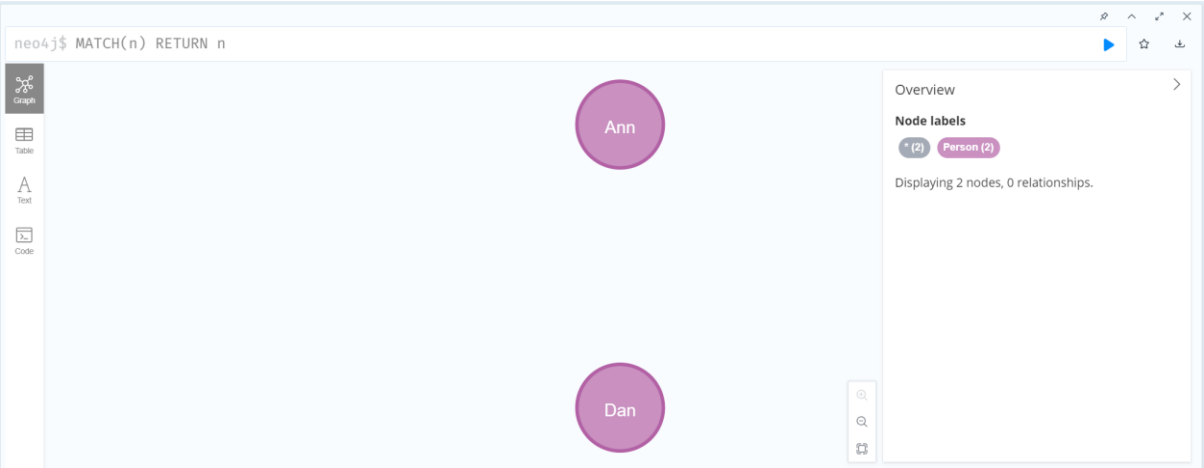
Enter the following Cypher statement in the sandbox:

CREATE
(:Person {name:'Dan', born: 'May 29, 1970', twitter: '@dan'}),
(:Person {name:'Ann', born: 'Dec 5, 1975'})

Run the statement on the Neo4j command line:



Now query all nodes with: **MATCH (n) RETURN n**



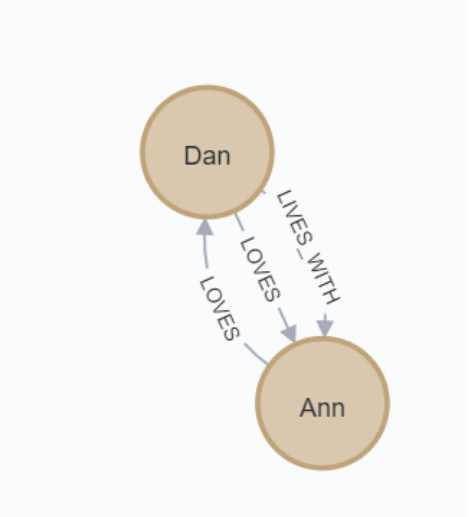
Now we are going to create the following relationships:

MATCH (pdan: Person), (pann: Person)
WHERE pdan.name = 'Dan' AND pann.name = 'Ann'
CREATE (pdan)-[:LOVES]->(pann)

MATCH (pdan: Person), (pann: Person)
WHERE pdan.name = 'Dan' AND pann.name = 'Ann'
CREATE (pann)-[:LOVES]->(pdan)

MATCH (pdan: Person), (pann: Person)
WHERE pdan.name = 'Dan' AND pann.name = 'Ann'
CREATE (pdan)-[:LIVES_WITH]->(pann)

This should give the following graph:



Now we want to know the person where Dan lives with:

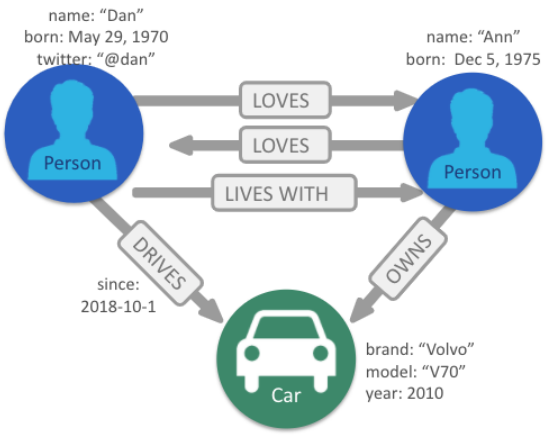
MATCH (p:Person {name : 'Dan'}) -[rel:LIVES_WITH]->(partner:Person)
RETURN partner.name

```
neo4j$ MATCH (p:Person {name : 'Dan'}) -[rel:LIVES_WITH]→(partner:Person) RETURN partner.name
```

Table | Text

	partner.name
1	"Ann"

Now extend the graph as follows:



Write the following queries:

1. Find all persons who own a Volvo car
2. Find the Car that is driven by Dan
3. Find all persons who love someone that owns a car.

Now create the following graph:

Nodes:

- 3 people: Alice, Bob, and Carol
- 2 companies: NeoTech and DataWorks

Relationships:

- Alice KNOWS Bob
- Bob KNOWS Carol
- Alice WORKS_AT NeoTech
- Bob WORKS_AT DataWorks
- Carol WORKS_AT NeoTech

Then create the following queries, and check if they work correctly:

1. Find who works at which company
2. Find all people who work at NeoTech
3. Find all people who Alice knows

Part 4

Suppose you need to store the following order in the database:

Ordernumber:122435
Orderdate 11/09/2021
Customer name: Frank Brown
Customer email: fbrown@gmail.com
Customer phone: 0623156543
Total price : 5160.00

quantity	Product number	Product name	price
2	A546	IPhone 12	980.00
4	S333	Samsung Galaxy 12S	800.00

1. Draw the tables including data that you need to store this order in a relational database.
2. Draw the collections including data that you need to store this order in a mongo database.
3. Draw the tables including data that you need to store this order in a cassandra database if you are interested in orders by customer.
4. Draw the database structure including data that you need to store this order in a neo4j database.

What to hand in?

1. A zip file of part 1
2. A screenshot of part 2
3. Multiple screenshots of part 3
4. A PDF of part 4