Astra DB

Big Data Systems

Dr. Rubi Boim

Motivation (for this course)

- Managing a cluster of Cassandra is not trivial
  Not the focus of this course

- Astra DB is a managed Cassandra DB service
  - Free version
  - We will use it to learn basic Cassandra programming

DataStax

- A commercial company established at 2010
  Cassandra was released at 2008

Provides

- Support for Cassandra
- DSE (DataStax Enterprise)
  own version of Cassandra with extra features
- Managed Database-as-a-service
  based on Apache Cassandra

DataStax

- A commercial company established at 2010
  Cassandra was released at 2008

IBM bought Datastax for $3B in 2025

Provides

- Support for Cassandra
- DSE (DataStax Enterprise)
  own version of Cassandra with extra features
- Managed Database-as-a-service
  based on Apache Cassandra

DataStax Enterprise (DSE)

- Commercial product - NOT open source
  an extension of Cassandra

Provides extra features / tools on top of Cassandra

- Cluster management
- GraphQL support
- Analytics
- Search capabilities
- …

Astra DB

- A managed Cassandra database service
- Launched at 2020
- Support Cassandra +3.11

Astra DB can be configure to run on

- AWS
- GCP
- Azure

Create an account

https://astra.datastax.com/

Create an account

https://astra.datastax.com/

You do NOT need to enter credit card details (!!!)

IBM

Sign Up for Free

Sign Up with GitHub

Sign Up with Google

Sign Up with IBMid

OR

First Name

Last Name

Company Email

I agree to DataStax, an IBM Company's Privacy Notice, MSA and Terms of Service

Sign Up

Already have an account? Sign In

REAL-TIME AI PLATFORM

Bring the power of real-time data to your next Generative AI project

Langflow New

Langflow is an easy way to build from simple to complex AI applications. It is a low-code platform that allows you to integrate AI into everything you do.

Exceptional Scalability and Robust Reliability

Create real-time data applications with the languages and APIs you already use. Built for speed and scale using trusted Apache Cassandra®.

Stream Real-Time Data at Scale

Deliver apps with massive throughput, low latency, and elastic scalability from any cloud, anywhere in the world.

© IBM Corp. 2025

DataStax

Register

Sign Up With GitHub

Sign Up With Google

OR

First Name

Last Name

Work Email

Password

I agree to the DataStax MSA, including the Astra Supplement.

Create Account

Already have an account? Sign In

Astra DB

Multi-cloud DBaaS Built on Apache Cassandra™

- Start in minutes. No credit card required. Up to 80 GB free monthly.
- Build faster with REST, GraphQL, CQL and JSON/Document APIs.
- Deploy multi-tenant or dedicated databases on AWS, Azure, or GCP.

Introducing Astra Streaming Beta!

Astra Streaming is an open, multi-cloud event-streaming and data-streaming cloud service powered by Apache Pulsar™, now available within Astra when you sign up for an account.

Before the “GEN AI” :)

Welcome to Astra

Knowing you better will help us improve your experience.

How do you plan to use Astra?

Work | Personal | Both | Other

Academia

How would you describe your role?

App Developer | AI Engineer | ML Engineer | Data Engineer | Cloud Engineer

Architect | Manager/Director | CTO/VP | Student or Learner | Other

What is your primary programming language?

Python | JS JavaScript/TypeScript | Java | GO Go | php PHP

Other | I don't code

How did you hear about Astra?

Google/Search | Blog/Tech Article | LinkedIn | Referred by someone | Twitter/X

YouTube | Partner/Integration | Workshop | Event/Conference | Other

TAU

Continue to Astra →

boim@cs.tau.ac.il Upgrade

Search Astra ⌘K

Docs Settings

Astra Databases

Manage your Astra databases. Add integrations, or upload and explore data.

No databases

Create a database to get started

Create database

Get started in code using the Astra CLI:

View CLI docs

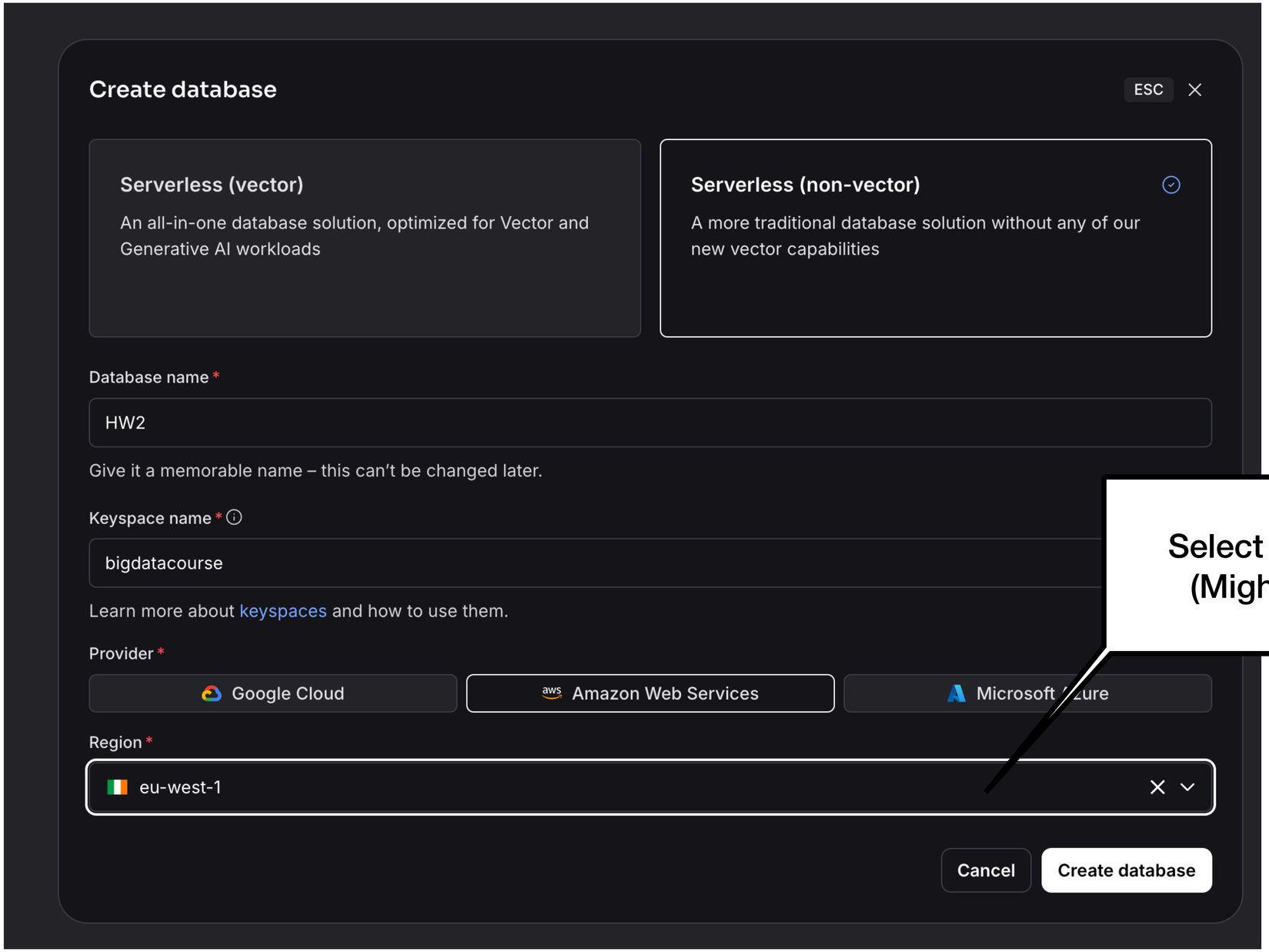
```
brew install datastax/astra-cli/astra-cli
```

```
astra setup --token APPLICATION_TOKEN
```

```
astra db create demo
```

```
astra db list-regions-serverless -c aws
```

```
astra user invite USER_EMAIL
```



Select the **free region**
(Might be different)

Astra Databases

Manage your Astra databases. Add integrations, or upload and explore data.

Create database

Serverless database usage totals

Read requests	Write requests	Storage consumed	Data transfer	Vector write dimensions	Vector read dimensions
0	0	0.00	0.00	0	0

Search databases...

Comfortable | Condensed

Name	Database ID	Reads	Writes	Storage	Data transfer	Vector rea...	Vector writ...	Created	Last used	Status
HW2	2db24957-...	0	0	0.00	0.00	0	0	12/21/2025	12/21/2025	Pending

Can take a few minutes...

Page Size: 10 Page 1 of 1

← Databases

HW2

Active

ID 2db24957-aa51-4125-ab30-9c9023f71a74 • API_ENDPOINT AWS eu-west-1 https://2db24957-aa51-4125-a...

Load Data | Connect

Overview | Health | Connect | CQL Console | CDC | Settings

Connect to your CQL Console

Use the CQL console to interact with your database through Cassandra Query Language (CQL).

Download Command Line Shell | eu-west-1

Region: eu-west-1

```
Connected as boim@cs.tau.ac.il.
Connected to cndb at cassandra.ingress:9042.
[cqlsh 6.8.0 | Cassandra 4.0.11.0-1e2868c88670 | CQL spec 3.4.5 | Native protocol v4 | TLS]
Use HELP for help.
token@cqlsh> describe keyspaces;

data_endpoint_auth  datastax_sla   system_traces
system_auth         bigdatacourse  system_views
system_schema       system         system_virtual_schema

token@cqlsh>
```

← Databases

HW2

Active

ID 2db24957-aa51-4125-ab30-9c9023f71a74 • API_ENDPOINT AWS eu-west-1 https://2db24957-aa51-4125-a...

Load Data | Connect

Overview | Health | Connect | CQL Console | CDC | Settings

Connect to your CQL Console

Use the CQL console to interact with your database through Cassandra Query Language (CQL).

Download Command Line Shell | eu-west-1

Region: eu-west-1

```
token@cqlsh:bigdatacourse> select * from users_by_country;

 country | user_id | age | name
---------+---------+-----+------

(0 rows)
token@cqlsh:bigdatacourse> desc users_by_country;

CREATE TABLE bigdatacourse.users_by_country (
    country text,
    user_id text,
    age int,
    name text,
    PRIMARY KEY (country, user_id)
) WITH CLUSTERING ORDER BY (user_id ASC)
    AND additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';

token@cqlsh:bigdatacourse>
```

See create code on the next slides

Bloom filter, gc_grace_seconds, …

Useful commands

- Autocomplete with the “tab” key
- `describe keyspaces;  (might work with “desc”)`
- `use <keyspace>`
- `describe tables`
- `describe <table>`
- `clear / CTRL+L (clear screen)`

- `create table / select / ...`

Demo - Create Table

```
CREATE TABLE users_by_country (
   country          TEXT,
   user_id          TEXT,
   name             TEXT,
   age              INT,
   PRIMARY KEY ((country), user_id)
);
```

users_by_country	
country	K
user_id	▲C
name	
age	

Demo - Insert

users_by_country	
country	K
user_id	▲C
name	
age	

```
INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('Israel','user_123','Rubi Boim', 22);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('USA','user_123','Rubi Boim', 22);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('Israel','user_124','Tova Milo', 18);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('USA','user_125','Lebron James',35);

INSERT INTO users_by_country(country, user_id, name)
VALUES ('USA','user_125','Michael Jordan');
```

Demo - Insert

users_by_country	
country	K
user_id	▲C
name	
age	

```
INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('Israel','user_123','Rubi Boim', 22);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('USA','user_123','Rubi Boim', 22);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('Israel','user_124','Tova Milo', 18);

INSERT INTO users_by_country(country, user_id, name, age)
VALUES ('USA','user_125','Lebron James',35);

INSERT INTO users_by_country(country, user_id, name)
VALUES ('USA','user_125','Michael Jordan');
```

What is Michael's age?

Demo - Select

Which queries are valid?

users_by_country	
country	K
user_id	▲C
name	
age	

```
SELECT * FROM users_by_country;

SELECT * FROM users_by_country WHERE country = 'Israel';

SELECT * FROM users_by_country WHERE name = 'Rubi Boim';

SELECT * FROM users_by_country WHERE user_id = 'user_123';

SELECT * FROM users_by_country WHERE country = 'Israel' AND user_id =
'user_123';

SELECT * FROM users_by_country WHERE country = 'Israel' AND name =
'Rubi Boim';
```

Demo - Select

Which queries are valid?

users_by_country	
country	K
user_id	▲C
name	
age	

```
SELECT * FROM users_by_country;

SELECT * FROM users_by_country WHERE country = 'Israel';

SELECT * FROM users_by_country WHERE name = 'Rubi Boim';

SELECT * FROM users_by_country WHERE user_id = 'user_123';

SELECT * FROM users_by_country WHERE country = 'Israel' AND user_id =
'user_123';

SELECT * FROM users_by_country WHERE country = 'Israel' AND name =
'Rubi Boim';
```

Remember - “ALLOW FILTERING is ANTI PATTERN

Demo - Update

UPDATE is similar to INSERT (*)

If you prefer to use the UPDATE syntax, “read the…”

Demo - Delete

Which queries are valid?

users_by_country	
country	K
user_id	▲C
name	
age	

```
DELETE FROM users_by_country WHERE name = 'Rubi Boim';

DELETE FROM users_by_country WHERE user_id = 'user_123';

DELETE FROM users_by_country WHERE country = 'Israel';

DELETE FROM users_by_country WHERE country = 'Israel'
AND name = 'Rubi Boim';

DELETE FROM users_by_country WHERE country = 'Israel'
AND user_id = 'user_123';
```

Demo - Drop table

```
DROP TABLE <table_name>;
```