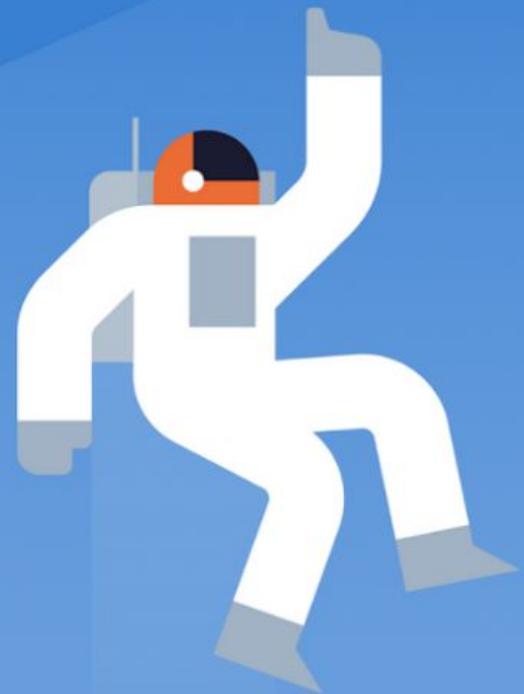


**DataStax**

# Cassandra Certification Workshop



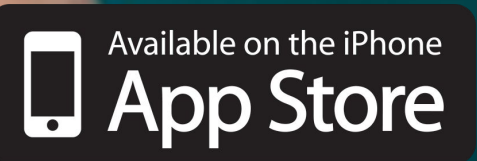
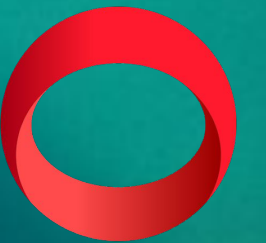
**LEVEL  
UP**

with the

**DataStax**

**Developers**

# menti.com



<https://github.com/DataStax-Academy/workshop-cassandra-certification>





# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. DS210 (Admin) practice
5. DS220 (Data Modeling) practice
6. Resources



# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. DS210 (Admin) practice
5. DS220 (Data Modeling) practice
6. Resources



# Which certification should I get?



*Designed for professionals who install, configure, manage and tune the performance of Apache Cassandra clusters*

**database administrators**  
**DevOps engineers**  
**Site Reliability Engineers (SREs)**



*Designed for professionals that use Apache Cassandra clusters to manage data*

**application developers**  
**data architects**  
**database designers**  
**database administrators**

# What resources do I have?



**Web:** [www.datastax.com/dev/certifications](http://www.datastax.com/dev/certifications)

DATASTAX

Products

Success

Learn

TRY FOR FREE

Developers > DataStax Certifications

DATASTAX

DATASTAX Certifications

Get certified as an Apache Cassandra Developer or Administrator

CERTIFICATION DETAILS

RULES AND PROCESS

TAKE AN EXAM

ADMINISTRATOR  
CERTIFICATION

ASSOCIATE LEVEL

DEVELOPER  
CERTIFICATION

ASSOCIATE LEVEL

Why should you become a DataStax Certified professional?

The demand for Apache Cassandra™ and NoSQL skills is skyrocketing, with Cassandra developers commanding **31% higher salaries**, the highest of any database technology.

**Github:** [DataStax-Academy/workshop-cassandra-certification](https://github.com/DataStax-Academy/workshop-cassandra-certification)

DataStax-Academy / workshop-cassandra-certification

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

main

1 branch

0 tags

Go to file

Add file

Code

Gitpod

drchung5

Update PRACTICE.md

6484b65 · 2 hours ago · 64 commits

images

Add files via upload

4 hours ago

PRACTICE-ADMIN.md

Create PRACTICE-ADMIN.md

yesterday

PRACTICE-DEV.md

Create PRACTICE-DEV.md

yesterday

PRACTICE.md

Update PRACTICE.md

2 hours ago

README.md

Minor formatting update

11 days ago

README.md

Apache Cassandra™ Certification Workshop

license Apache-2.0 chat 215 online

**Training:** [academy.datastax.com](https://academy.datastax.com)

Resume

Nothing to resume yet

My Courses

See courses you are enrolled in

Catalog

See a complete list of available courses

Resources

Browse or download

Catalog

ADMIN

3 LEARNING PATH

DataStax Academy Apache Cassandra™ Administrator Curriculum

Enroll

DEVELOPER

3 LEARNING PATH

DataStax Academy Apache Cassandra™ Developer Path Curriculum

Enroll

KAFKA

CONNECTOR

DataStax Apache Kafka™ Connector Online Course

Enroll

DATASTAX

CERTIFICATION

DataStax Certification Online Course

Completed

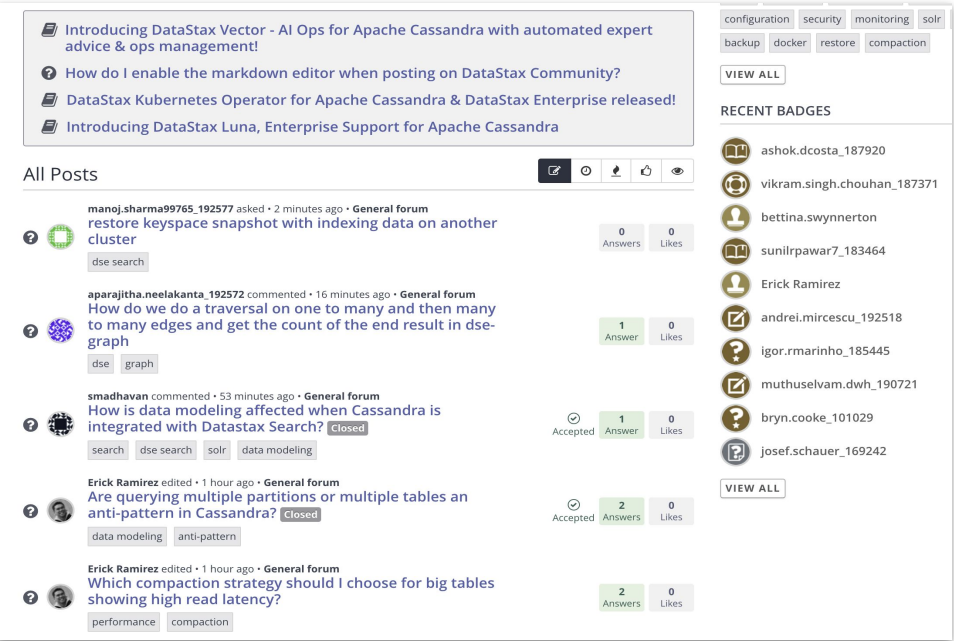
DS101

Apache

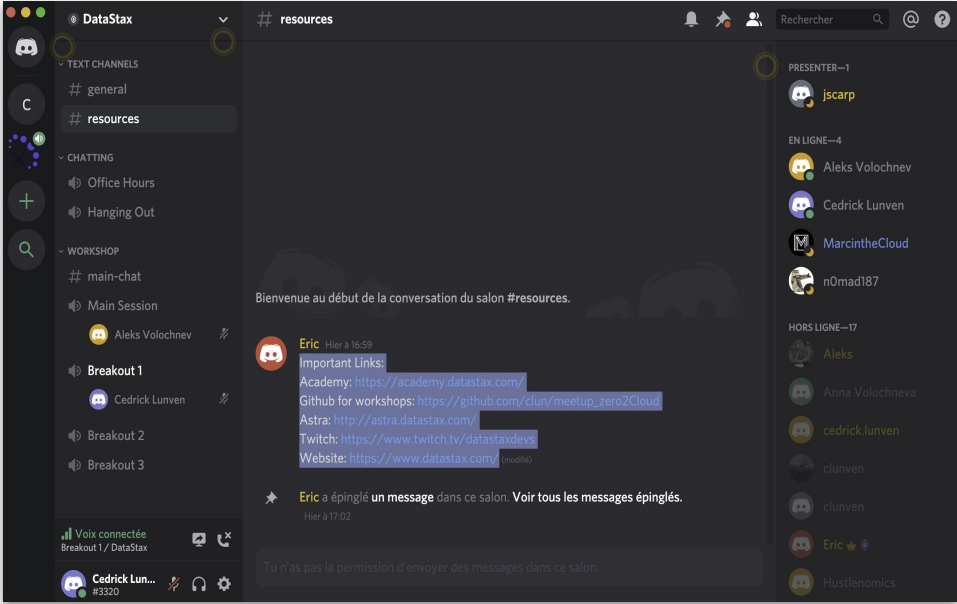
Online Course

**DataStax Developers**

**Forum:** [community.datastax.com](https://community.datastax.com)



**Chat:** [bit.ly/cassandra-workshop](https://bit.ly/cassandra-workshop)



**Discord**

# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. DS210 (Admin) practice
5. DS220 (Data Modeling) practice
6. Resources



# Step 1

Go to

<https://www.datastax.com/dev/certifications>,

read through the material, and take special note of the **Exam Rules and Process** section.

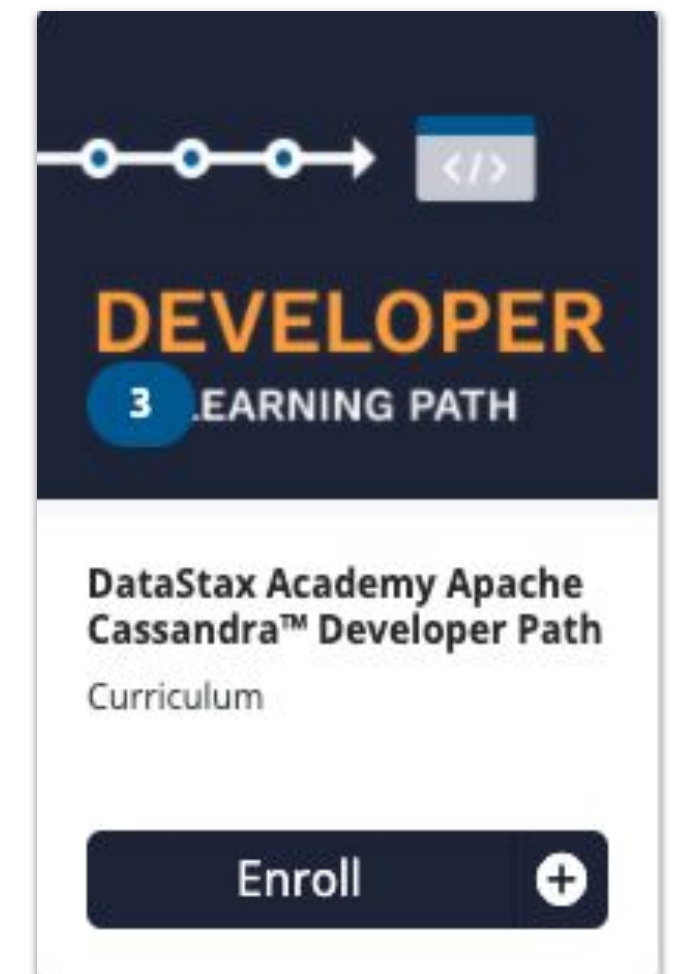
**Exams are proctored.**



## Step 2



Choose a learning path,  
either the  
**Administrator Certification** or the  
**Developer Certification.**



## Step 3

**Go to DataStax Academy and sign up if you have not already done so. Academy is FREE along with all of the course content.**

## Step 4

Based on the **learning path** you've chosen complete the course material within Academy.



The card features a dark blue header with a progress indicator (three circles and an arrow) and two gear icons. Below the header, the word "ADMIN" is in large orange letters, followed by "3 LEARNING PATH" in a blue circle and white text. The main body is white with the text "DataStax Academy Apache Cassandra™ Administrator Curriculum". At the bottom is a dark blue "Enroll" button with a white plus icon. The footer is white with the text "DS201 & DS210" in orange.

**ADMIN**  
3 LEARNING PATH

DataStax Academy Apache  
Cassandra™ Administrator  
Curriculum

Enroll +

**DS201 & DS210**

These links are provided for you in the **Learning Paths** section at



The card features a dark blue header with a progress indicator (three circles and an arrow) and a code icon. Below the header, the word "DEVELOPER" is in large orange letters, followed by "3 LEARNING PATH" in a blue circle and white text. The main body is white with the text "DataStax Academy Apache Cassandra™ Developer Path Curriculum". At the bottom is a dark blue "Enroll" button with a white plus icon. The footer is white with the text "DS201 & DS220" in orange.

**DEVELOPER**  
3 LEARNING PATH

DataStax Academy Apache  
Cassandra™ Developer Path  
Curriculum

Enroll +

**DS201 & DS220**

<https://www.datastax.com/dev/certifications>

## Step 5



**Get your exam voucher.**

**Complete a learning path and  
email [academy@datastax.com](mailto:academy@datastax.com).**

**OR**



## Step 5

DataStax

**Astra DB**

**Create an Astra database per instructions from:**

**<https://github.com/DataStax-Academy/workshop-crud-with-python-and-node>**

**Using the same email you registered to eventbrite**

## Step 6

**Take your exam.**

**Don't forget to** 

**Full details are at**

**<https://github.com/DataStax-Academy/workshop-cassandra-certification>**

# Demo the process



# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. **DS201 (Foundation) practice**
4. DS210 (Admin) practice
5. DS220 (Data Modeling) practice
6. Resources





# 1. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statements:

```
CREATE TABLE roller_coasters (
```

```
    name TEXT,
```

```
    park TEXT,
```

```
    rating INT,
```

```
    PRIMARY KEY((name))
```

```
);
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Millenium Force', 'Cedar Point', 8 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Formula Rossa', 'Ferrari World', 9 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Steel Dragon 2000', 'Nagashima Spa Land', 10 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Millenium Force', 'Cedar Point', 7 );
```

How many rows will the roller\_coasters table have after executing all the CQL statements?

A. none

B. 2

C. 3

D. 4

# 1. Solution

Consider the CQL statements:

```
CREATE TABLE roller_coasters (
```

```
    name TEXT,
```

```
    park TEXT,
```

```
    rating INT,
```

```
    PRIMARY KEY((name))
```

```
);
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Millenium Force', 'Cedar Point', 8 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Formula Rossa', 'Ferrari World', 9 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Steel Dragon 2000', 'Nagashima Spa Land', 10 );
```

```
INSERT INTO roller_coasters (name, park, rating)
```

```
VALUES ('Millenium Force', 'Cedar Point', 7 );
```

How many rows will the roller\_coasters table have after executing all the CQL statements?

A. none

B. 2

C. 3

D. 4

The **first** and **fourth** INSERTS use the **same primary key** so they cause an **upsert**.

Therefore only 3 rows are created.

## 2. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statements:

```
CREATE TABLE songs (  
    artist TEXT,  
    title TEXT,  
    length_seconds INT,  
    PRIMARY KEY((artist, title))  
);
```

```
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('The Beatles', 'Yesterday', 123 );
```

```
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('The Beatles', 'Let It Be', 243 );
```

```
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('Abba', 'Fernando', 255 );
```

```
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('Frank Sinatra', 'Yesterday', 235 );
```

What is the result of executing all the CQL statements?

- A. A table with 1 partition.
- B. A table with 2 partitions.
- C. A table with 3 partitions.
- D. A table with 4 partitions.

## 2. Solution

Consider the CQL statements:

```
CREATE TABLE songs (  
    artist TEXT,  
    title TEXT,  
    length_seconds INT,  
    PRIMARY KEY((artist, title))  
);  
  
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('The Beatles', 'Yesterday', 123 );  
  
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('The Beatles', 'Let It Be', 243 );  
  
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('Abba', 'Fernando', 255 );  
  
INSERT INTO songs (artist, title, length_seconds)  
VALUES ('Frank Sinatra', 'Yesterday', 235 );
```

What is the result of executing all the CQL statements?

A. A table with 1 partition.

B. A table with 2 partitions.

C. A table with 3 partitions.

**D. A table with 4 partitions.**

The partition key consists of artist and title (incidentally, it is also the whole primary key).

Each INSERT has a unique artist/title pair so there are no upserts and each INSERT results in a unique partition.



# 3. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statement:

```
CREATE TABLE cars (  
    make TEXT,  
    model TEXT,  
    year INT,  
    color TEXT,  
    cost INT,  
    PRIMARY KEY ((make, model), year, color)  
);
```

Which of the following is a valid query for the cars table?

- A.  
SELECT \* FROM cars  
WHERE make='Ford';
- B.  
SELECT \* FROM cars  
WHERE year = 1969  
AND color = 'Red';
- C.  
SELECT \* FROM cars  
WHERE make='Ford'  
AND model = 'Mustang'  
AND year = 1969;
- D.  
SELECT \* FROM cars  
WHERE make='Ford'  
AND model = 'Mustang'  
AND color = 'Red';

# 3. Solution

Consider the CQL statement:

```
CREATE TABLE cars (  
    make TEXT,  
    model TEXT,  
    year INT,  
    color TEXT,  
    cost INT,  
    PRIMARY KEY ((make, model), year, color)  
);
```

The partition key consists of make and model so A and B are excluded because the **WHERE clause does not include the partition key.**

C and D both include the partition key **but clustering columns can only be constrained L-R in the order they appear in the primary key.**

Since year appears before color, C is correct and D is excluded.

Which of the following is a valid query for the cars table?

A.  
SELECT \* FROM cars  
WHERE **make='Ford'**;

B.  
SELECT \* FROM cars  
WHERE **year = 1969**  
AND **color = 'Red'**;

C.  
SELECT \* FROM cars  
WHERE **make='Ford'**  
AND **model = 'Mustang'**  
AND **year = 1969;**

D.  
SELECT \* FROM cars  
WHERE **make='Ford'**  
AND **model = 'Mustang'**  
AND **color = 'Red'**;

# 4. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statement:

```
CREATE TABLE employees (  
  id TEXT,  
  name TEXT,  
  department TEXT,  
  PRIMARY KEY ((id))  
);
```

```
CREATE TABLE employees_by_department (  
  id TEXT,  
  name TEXT,  
  department TEXT,  
  PRIMARY KEY ((department), id)  
);
```

```
BEGIN BATCH  
  INSERT INTO employees (id, name, department)  
    VALUES ('AC1123', 'Joe', 'legal');  
  
  INSERT INTO employees_by_department (id, name, department)  
    VALUES ('AC1123', 'Joe', 'legal');  
APPLY BATCH;
```

What is a valid statement about this batch?

- A. It is a single-partition batch that can be applied.
- B. It is a single-partition batch that cannot be applied.
- C. It is a multi-partition batch that can be applied.
- D. It is a multi-partition batch that cannot be applied.

## 4. Solution

Consider the CQL statement:

```
CREATE TABLE employees (  
  id TEXT,  
  name TEXT,  
  department TEXT,  
  PRIMARY KEY ((id))  
);  
  
CREATE TABLE employees_by_department (  
  id TEXT,  
  name TEXT,  
  department TEXT,  
  PRIMARY KEY ((department), id)  
);  
  
BEGIN BATCH  
  INSERT INTO employees (id, name, department)  
    VALUES ('AC1123', 'Joe', 'legal');  
  
  INSERT INTO employees_by_department (id, name, department)  
    VALUES ('AC1123', 'Joe', 'legal');  
APPLY BATCH;
```

What is a valid statement about this batch?

- A. It is a single-partition batch that can be applied.
- B. It is a single-partition batch that cannot be applied.
- C. It is a multi-partition batch that can be applied.**
- D. It is a multi-partition batch that cannot be applied.

The two INSERTS are into different tables which makes them different partitions.

Even if one or both result in upserts there is nothing preventing this batch from being applied.

# 5. CQL – Developer and Administrator Exams (DS201)

Consider the table definition with a primary key

omitted:

```
CREATE TABLE reviews_by_restaurant (  
  name TEXT,  
  city TEXT,  
  reviewer TEXT,  
  rating INT,  
  comments TEXT,  
  review_date TIMEUUID,  
  PRIMARY KEY (...)  
);
```

It is known that:

- Restaurant Reviews are uniquely identified by a combination of name, city and reviewer
- Restaurant Reviews are retrieved from the table using combination of name, city
- The table has multi-row partitions

What primary key does this table have?

- A. PRIMARY KEY((name), reviewer, city)
- B. PRIMARY KEY((name, city), reviewer)
- C. PRIMARY KEY((name, reviewer), city)
- D. PRIMARY KEY(reviewer, name, city)



# 5. Solution

Consider the table definition with a primary key omitted:

```
CREATE TABLE reviews_by_restaurant (  
  name TEXT,  
  city TEXT,  
  reviewer TEXT,  
  rating INT,  
  comments TEXT,  
  review_date TIMEUUID,  
  PRIMARY KEY (...)  
);
```

Since restaurant reviews are uniquely identified by a combination of name, city and reviewer the primary key must include all three fields.

Since restaurant reviews are retrieved from the table using combination of name, city, these two fields must come before reviewer in the primary key.

*A primary key ((name), city, reviewer) would also have worked fine, with the query selecting a (contiguous) part of a wider partition.*

It is known that:

- Restaurant Reviews are uniquely identified by a combination of name, city and reviewer
- Restaurant Reviews are retrieved from the table using combination of name, city
- The table has multi-row partitions

What primary key does this table have?

A. PRIMARY KEY((name), reviewer, city)

**B. PRIMARY KEY((name, city), reviewer)**

C. PRIMARY KEY((name, reviewer), city)

D. PRIMARY KEY(reviewer, name, city)

## 6. CQL – Developer Exam (DS220)

Consider the table definition and the CQL query:

```
CREATE TABLE teams (  
    name TEXT PRIMARY KEY,  
    wins INT,  
    losses INT,  
    ties INT  
);
```

```
SELECT * FROM teams_by_wins WHERE wins = 4;
```

Which materialized view definition can be used to support the query?

A.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
teams_by_wins AS  
SELECT * FROM teams  
PRIMARY KEY((name), wins);
```

B.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
teams_by_wins AS  
SELECT * FROM teams  
PRIMARY KEY((wins), name);
```

C.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
teams_by_wins AS  
SELECT * FROM teams  
WHERE name IS NOT NULL AND wins IS NOT NULL  
PRIMARY KEY((name), wins);
```

D.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
teams_by_wins AS  
SELECT * FROM teams  
WHERE wins IS NOT NULL AND name IS NOT NULL  
PRIMARY KEY((wins), name);
```

## 6. Solution

Consider the table definition and the CQL query:

```
CREATE TABLE teams (  
  name TEXT PRIMARY KEY,  
  wins INT,  
  losses INT,  
  ties INT  
);
```

```
SELECT * FROM teams_by_wins WHERE wins = 4;
```

Since **primary key fields cannot be NULL** the WHERE clause **must include a NULL check**.

Since the **WHERE clause in the SELECT is based on wins**, **wins must be the partition key**.

Which materialized view definition can be used to support the query?

A.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
  teams_by_wins AS  
  SELECT * FROM teams  
  PRIMARY KEY((name), wins);
```

B.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
  teams_by_wins AS  
  SELECT * FROM teams  
  PRIMARY KEY((wins), name);
```

C.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
  teams_by_wins AS  
  SELECT * FROM teams  
  WHERE name IS NOT NULL AND wins IS NOT NULL  
  PRIMARY KEY((name), wins);
```

D.

```
CREATE MATERIALIZED VIEW IF NOT EXISTS  
  teams_by_wins AS  
  SELECT * FROM teams  
  WHERE wins IS NOT NULL AND name IS NOT NULL  
  PRIMARY KEY((wins), name);
```

# 7. CQL – Developer Exam (DS220)

Consider the table definition and the CQL query:

```
CREATE TABLE restaurants_by_city (  
  name TEXT,  
  city TEXT,  
  cuisine TEXT,  
  price int,  
  PRIMARY KEY ((city), name)  
);
```

```
SELECT * FROM restaurants_by_city  
WHERE city = 'Sydney'  
AND cuisine = 'sushi';
```

Which secondary index can be used to support the query?

- A.  
CREATE INDEX cuisine\_restaurants\_by\_city\_2i  
ON restaurants\_by\_city (cuisine);
- B.  
CREATE INDEX cuisine\_restaurants\_by\_city\_2i  
ON restaurants\_by\_city (city, cuisine);
- C.  
CREATE INDEX cuisine\_restaurants\_by\_city\_2i  
ON restaurants\_by\_city (cuisine, city);
- D.  
CREATE INDEX cuisine\_restaurants\_by\_city\_2i  
ON restaurants\_by\_city (city, name, cuisine);

# 7. Solution

Consider the table definition and the CQL query:

```
CREATE TABLE restaurants_by_city (  
  name TEXT,  
  city TEXT,  
  cuisine TEXT,  
  price int,  
  PRIMARY KEY ((city), name)  
);
```

```
SELECT * FROM restaurants_by_city  
WHERE city = 'Sydney'  
AND cuisine = 'sushi';
```

B, C, and D are incorrect because **indexes on multiple columns are not supported**.

Which secondary index can be used to support the query?

A.

```
CREATE INDEX cuisine_restaurants_by_city_2i  
ON restaurants_by_city (cuisine);
```

B.

```
CREATE INDEX cuisine_restaurants_by_city_2i  
ON restaurants_by_city (city, cuisine);
```

C.

```
CREATE INDEX cuisine_restaurants_by_city_2i  
ON restaurants_by_city (cuisine, city);
```

D.

```
CREATE INDEX cuisine_restaurants_by_city_2i  
ON restaurants_by_city (city, name, cuisine);
```

## 8. CQL – Developer and Administrator Exams (DS201)

Which statement describes the WHERE clause in a query?

- A. WHERE clauses must reference all the fields of the partition key.
- B. WHERE clauses must reference all the fields of the clustering key.
- C. WHERE clauses must reference all the fields of the primary key.
- D. WHERE clauses must reference all the fields of the partition key and clustering key.



## 8. Solution

Which statement describes the WHERE clause in a query?

- A. WHERE clauses must reference all the fields of the partition key.
- B. WHERE clauses must reference all the fields of the clustering key.
- C. WHERE clauses must reference all the fields of the primary key.
- D. WHERE clauses must reference all the fields of the partition key and clustering key.

Only the fields of the partition key are required.

# 9. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statements:

```
CREATE TYPE NAME (  
    first TEXT,  
    last TEXT  
);
```

```
CREATE TABLE people (  
    id UUID,  
    name NAME,  
    email TEXT,  
    PRIMARY KEY((id), email)  
);
```

Which INSERT statement can be used to insert a row in the people table?

A.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), {first:'foo', last:'bar'}, 'foo@datastax.com' );
```

B.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), name: {'foo', 'bar'}, 'foo@datastax.com' );
```

C.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), 'foo', 'bar', 'foo@datastax.com' );
```

D.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), ('foo', 'bar'), 'foo@datastax.com' );
```

# 9. Solution

Consider the CQL statements:

```
CREATE TYPE NAME (  
    first TEXT,  
    last TEXT  
);  
  
CREATE TABLE people (  
    id UUID,  
    name NAME,  
    email TEXT,  
    PRIMARY KEY((id), email)  
);
```

The **fields** of the **user defined type** are passed using **JSON**.

Which INSERT statement can be used to insert a row in the people table?

A.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), {first:'foo', last:'bar'}, 'foo@datastax.com' );
```

B.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), name: {'foo', 'bar'}, 'foo@datastax.com' );
```

C.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), 'foo', 'bar', 'foo@datastax.com' );
```

D.

```
INSERT INTO people (id, name, email)  
VALUES (UUID(), ('foo', 'bar'), 'foo@datastax.com' );
```

# 10. CQL – Developer and Administrator Exams (DS201)

Consider the CQL statements:

```
CREATE TABLE emails_by_user (  
  username TEXT,  
  email TEXT,  
  description TEXT,  
  nickname TEXT STATIC,  
  PRIMARY KEY((username), email)  
);  
  
INSERT INTO emails_by_user (username, email, description, nickname)  
VALUES ('dc1234', 'david@datastax.com', 'work', 'Dave');  
  
INSERT INTO emails_by_user (username, email, description, nickname)  
VALUES ('dc1234', 'david@gmail.com', 'personal', 'Dave');  
  
UPDATE emails_by_user SET nickname = 'Davey', description = 'school'  
WHERE username = 'dc1234' AND email = 'david@gmail.com';  
  
SELECT * FROM emails_by_user WHERE username = 'dc1234';
```

What is the result of executing theses CQL statements?

- A.

username	email	nickname	description
dc1234	david@datastax.com	Dave	work
dc1234	david@gmail.com	Davey	school
- B.

username	email	nickname	description
dc1234	david@datastax.com	Davey	work
dc1234	david@gmail.com	Davey	school
- C.

username	email	nickname	description
dc1234	david@gmail.com	Davey	school
- D.

username	email	nickname	description
dc1234	david@datastax.com	Dave	work

# 10. Solution

Consider the CQL statements:

```
CREATE TABLE emails_by_user (  
  username TEXT,  
  email TEXT,  
  description TEXT,  
  nickname TEXT STATIC,  
  PRIMARY KEY((username), email)  
);  
  
INSERT INTO emails_by_user (username, email, description, nickname)  
VALUES ('dc1234', 'david@datastax.com', 'work', 'Dave');  
  
INSERT INTO emails_by_user (username, email, description, nickname)  
VALUES ('dc1234', 'david@gmail.com', 'personal', 'Dave');  
  
UPDATE emails_by_user SET nickname = 'Davey', description = 'school'  
WHERE username = 'dc1234' AND email = 'david@gmail.com';  
  
SELECT * FROM emails_by_user WHERE username = 'dc1234';
```

Because email is a clustering column the table has one partition with two rows.

The nickname field is static so it was set to Davey for the entire partition.

What is the result of executing theses CQL statements?

- A.

username	email	nickname	description
dc1234	david@datastax.com	Dave	work
dc1234	david@gmail.com	Davey	school
- B.

username	email	nickname	description
dc1234	david@datastax.com	Davey	work
dc1234	david@gmail.com	Davey	school
- C.

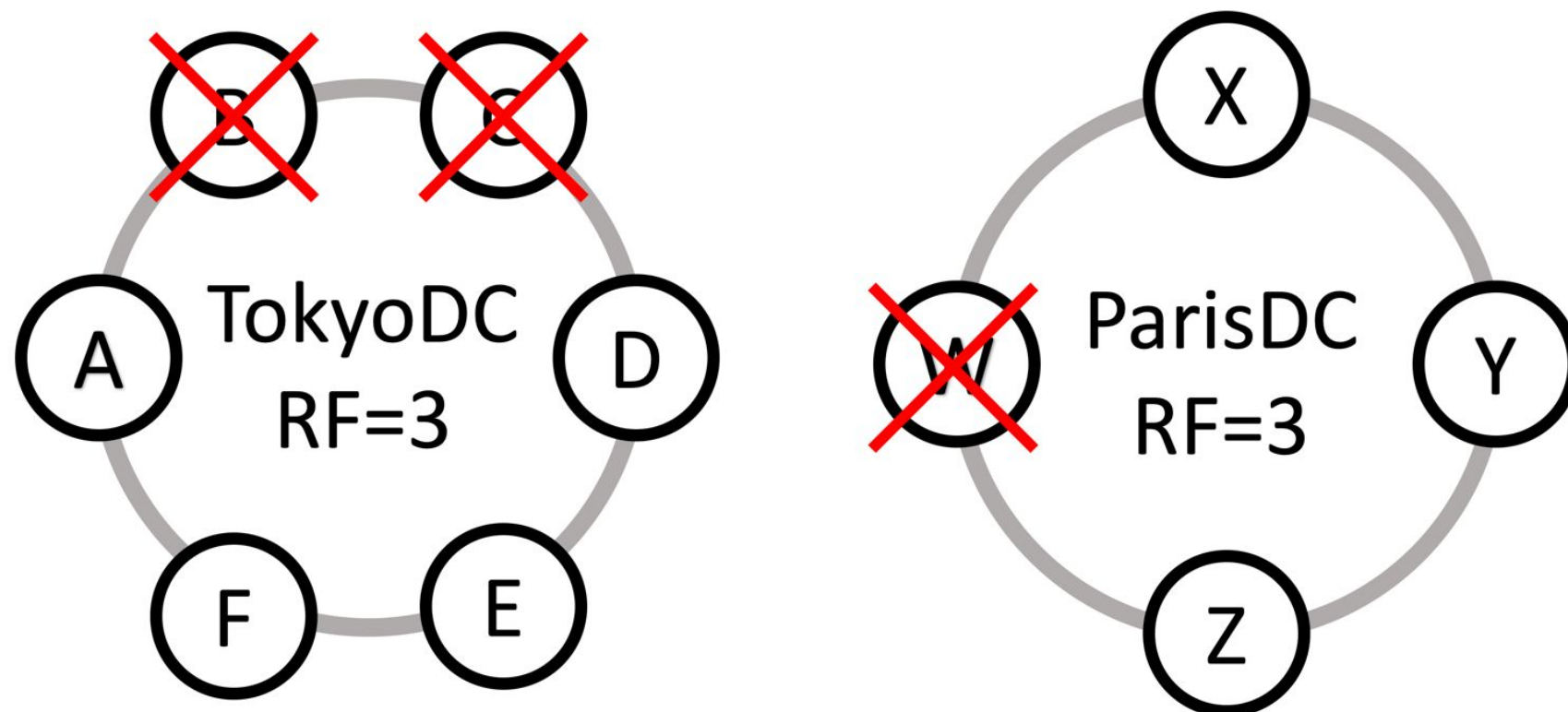
username	email	nickname	description
dc1234	david@gmail.com	Davey	school
- D.

username	email	nickname	description
dc1234	david@datastax.com	Dave	work

# 11. Architecture Exams (DS201)

Consider the two datacenters in the diagram.

TokyoDC has six nodes (two failed and four active) and a replication factor of 3, and ParisDC four nodes (one failed and three active) and a replication factor of 3.



What is a valid statement about a read request made at consistency level of LOCAL QUORUM to coordinator node Z in ParisDC?

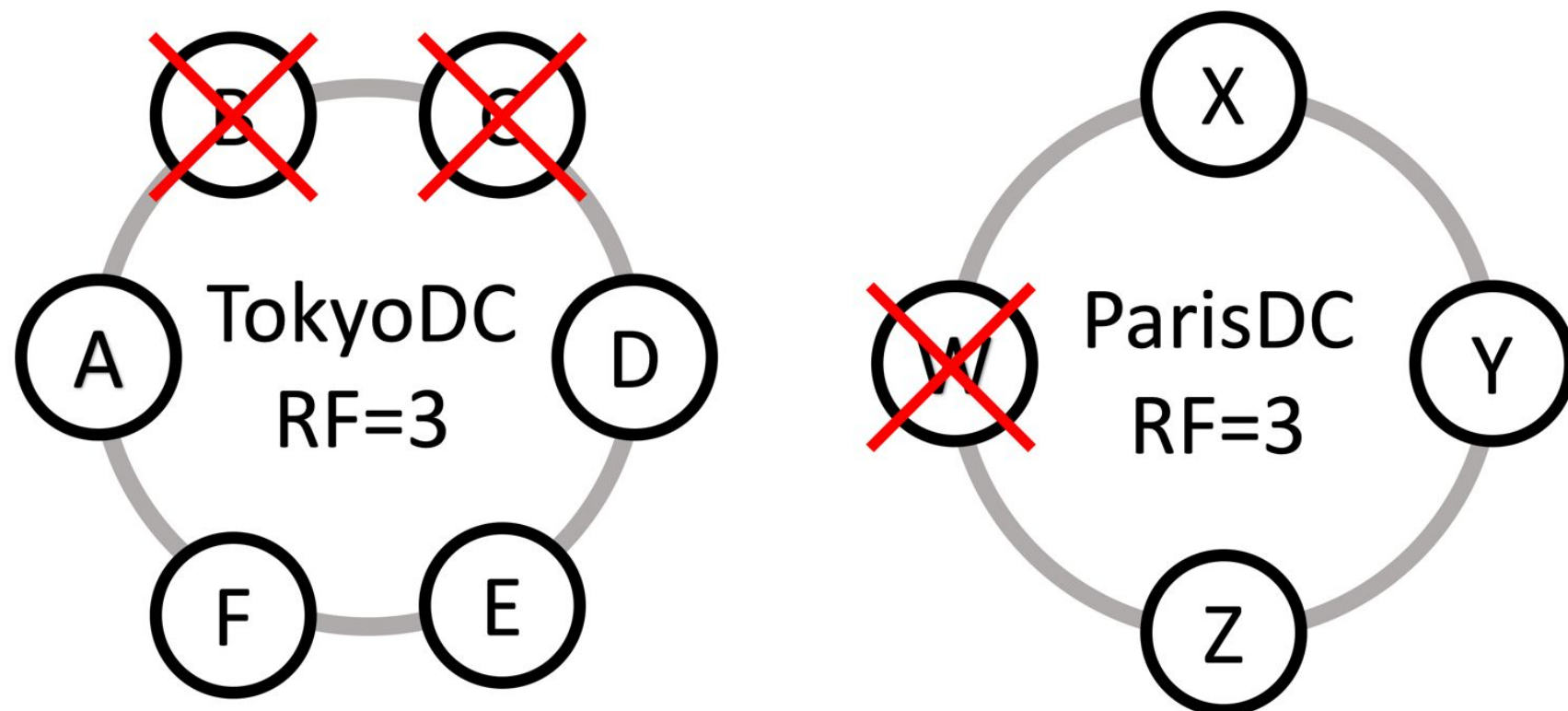
- A. The request will be handled in data center ParisDC and will fail.
- B. The request will be handled in data center ParisDC and will succeed.
- C. The request will be retried in data center TokyoDC and will fail.
- D. The request will be retried in data center TokyoDC and will succeed.



# 11. Solution

Consider the two datacenters in the diagram.

TokyoDC has six nodes (two failed and four active) and a replication factor of 3, and ParisDC four nodes (one failed and three active) and a replication factor of 3.



What is a valid statement about a read request made at consistency level of LOCAL QUORUM to coordinator node Z in ParisDC?

- A. The request will be handled in data center ParisDC and will fail.
- B. The request will be handled in data center ParisDC and will succeed.
- C. The request will be retried in data center TokyoDC and will fail.
- D. The request will be retried in data center TokyoDC and will succeed.

**LOCAL QUORUM** requires a quorum (more than half) of the replicas in a the local data center to respond in order to succeed.

Since only 1 of 4 nodes have failed there will be at least 2 replicas available to handle the request. 2 is the quorum of 3, therefore the request will succeed.

# 12. Architecture Exams (DS201)

Consider these CQL traces (You may need to scroll to see the entire trace.):

activity	timestamp	source	source_elapsed	client
Execute CQL3 query	2020-10-09 16:18:49.223000	10.52.26.153	0	10.52.13.186
Parsing INSERT INTO NAMES (id, name) VALUES (UUID(), 'Dave'); [CoreThread-0]	2020-10-09 16:18:49.223000	10.52.26.153	328	10.52.13.186
Preparing statement [CoreThread-0]	2020-10-09 16:18:49.223000	10.52.26.153	690	10.52.13.186
Determining replicas for mutation [CoreThread-0]	2020-10-09 16:18:49.224000	10.52.26.153	1834	10.52.13.186
Appending to commitlog [CoreThread-0]	2020-10-09 16:18:49.225000	10.52.26.153	2193	10.52.13.186
Adding to names memtable [CoreThread-0]	2020-10-09 16:18:49.225000	10.52.26.153	2326	10.52.13.186
Request complete	2020-10-09 16:18:49.225966	10.52.26.153	2966	10.52.13.186

At what elapsed time is the data persisted so that it will survive an unexpected node shutdown?

- A. 690 microseconds
- B. 1834 microseconds
- C. 2193 microseconds
- D. 2966 microseconds

# 12. Solution

Consider these CQL traces (You may need to scroll to see the entire trace.):

activity	timestamp	source	source_elapsed	client
Execute CQL3 query	2020-10-09 16:18:49.223000	10.52.26.153	0	10.52.13.186
Parsing INSERT INTO NAMES (id, name) VALUES (UUID(), 'Dave'); [CoreThread-0]	2020-10-09 16:18:49.223000	10.52.26.153	328	10.52.13.186
Preparing statement [CoreThread-0]	2020-10-09 16:18:49.223000	10.52.26.153	690	10.52.13.186
Determining replicas for mutation [CoreThread-0]	2020-10-09 16:18:49.224000	10.52.26.153	1834	10.52.13.186
Appending to commitlog [CoreThread-0]	2020-10-09 16:18:49.225000	10.52.26.153	2193	10.52.13.186
Adding to names memtable [CoreThread-0]	2020-10-09 16:18:49.225000	10.52.26.153	2326	10.52.13.186
Request complete	2020-10-09 16:18:49.225966	10.52.26.153	2966	10.52.13.186

At what elapsed time is the data persisted so that it will survive an unexpected node shutdown?

- A. 690 microseconds
- B. 1834 microseconds
- C. 2193 microseconds
- D. 2966 microseconds

Once data is written to commit log it will survive an unexpected node shutdown.

# 13. Architecture Exams (DS201)

How is Replication Factor configured in Cassandra?

- A. per cluster
- B. per keyspace
- C. per operation
- D. per node

# 13. Solution

How is Replication Factor configured in Cassandra?

A. per cluster

**B. per keyspace**

C. per operation

D. per node

Replication factor (and strategy) **MUST BE** configured when creating a keyspace.

# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. **DS210 (Admin) practice**
5. DS220 (Data Modeling) practice
6. Resources





# 14. Administrator Exams (DS210)

What are two options for internode\_encryption in Cassandra? (Choose two.)

A. client

B. node

C. rack

D. enabled

E. dc

# 14. Solution

What are two options for internode\_encryption in Cassandra? (Choose two.)

A. client

B. node

C. rack

D. enabled

E. dc

The available options are: all, none, **dc** and **rack**.

# 15. Administrator Exams (DS210)

Which configuration file is used to set garbage collection properties for Cassandra?

- A. cassandra.yaml
- B. jvm.options
- C. cassandra-env.sh
- D. gc.options

# 15. Solution

Which configuration file is used to set garbage collection properties for Cassandra?

A. `cassandra.yaml`

**B. `jvm.options`**

C. `cassandra-env.sh`

D. `gc.options`

The purpose of the `jvm.options` file is to put JVM-specific properties (like garbage collection) in one place.

# 16. Administrator Exams (DS210)

Consider the table definition and how a single row is stored in one Memtable and two SSTables on a

Cassandra node:

```
CREATE TABLE tests (  
  id INT PRIMARY KEY,  
  test TEXT,  
  score int  
);
```

Memtable

id: 11 timestamp: 1392353211  
score: 75 timestamp: 1392353211

SSTable

id: 11 timestamp: 1204596828  
test: math timestamp: 1204596828  
score: 62 timestamp: 1204596828

SSTable

id: 11 timestamp: 1183608357  
test: english timestamp: 1183608357  
score: 48 timestamp: 1183608357

What are the current values for this row?

A.  
id | test | score  
----+-----+-----  
11 | english | 48

B.  
id | test | score  
----+-----+-----  
11 | math | 75

C.  
id | test | score  
----+-----+-----  
11 | math | 62

D.  
id | test | score  
----+-----+-----  
11 | math | 48

# 16. Solution

Consider the table definition and how a single row is stored in one Memtable and two SSTables on a

Cassandra node:

```
CREATE TABLE tests (  
  id INT PRIMARY KEY,  
  test TEXT,  
  score int  
);
```

Memtable

```
id: 11 timestamp: 1392353211  
score: 75 timestamp: 1392353211
```

SSTable

```
id: 11 timestamp: 1204596828  
test: math timestamp: 1204596828  
score: 62 timestamp: 1204596828
```

SSTable

```
id: 11 timestamp: 1183608357  
test: english timestamp: 1183608357  
score: 48 timestamp: 1183608357
```

What are the current values for this row?

A.

```
id | test   | score  
---+-----+-----  
11 | english | 48
```

B.

```
id | test  | score  
---+-----+-----  
11 | math  | 75
```

C.

```
id | test  | score  
---+-----+-----  
11 | math  | 62
```

D.

```
id | test  | score  
---+-----+-----  
11 | math  | 48
```

Data for a row may be spread across the memtable and multiple SSTables. The row value is made up of the most recent (timestamp) value for each column.

# 17. Administrator Exams (DS210)

What is a valid statement about a coordinator node handling a query at consistency level THREE?

- A. The coordinator node sends a direct read request to all replicas.
- B. The coordinator node sends a direct read request to three replicas.
- C. The coordinator node sends a background read repair request to three replicas.
- D. The coordinator node sends a direct read request to one replica and digest requests to two replicas.



# 17. Solution

What is a valid statement about a coordinator node handling a query at consistency level THREE?

- A. The coordinator node sends a direct read request to all replicas.
- B. The coordinator node sends a direct read request to three replicas.
- C. The coordinator node sends a background read repair request to three replicas.
- D. The coordinator node sends a direct read request to one replica and digest requests to two replicas.

The **coordinator** node only **sends a direct read** request to **one node** and **sends digest request(s)** to the **remainder** necessary to meet the consistency level.

The **coordinator** node then **compares** the data read directly **with the digest(s)**. If they agree the result is returned to the client.

If they **do not agree** the **most recent timestamped result** is **considered current** and **sent to the client**. The coordinator node may need to request the latest timestamped version from a replica.

## 18. Administrator Exams (DS210)

What is a valid statement about a write made at consistency level LOCAL\_QUORUM against a keyspace with replication factor of 3?

- A. The coordinator node will send a write to one node.
- B. The coordinator node will send writes to two nodes.
- C. The coordinator node will send writes to three nodes.
- D. The coordinator node will send writes to all nodes.

## 18. Solution

What is a valid statement about a write made at consistency level LOCAL\_QUORUM against a keyspace with replication factor of 3?

- A. The coordinator node will send a write to one node.
- B. The coordinator node will send writes to two nodes.
- C. The coordinator node will send writes to three nodes.
- D. The coordinator node will send writes to all nodes.

The **coordinator** node **will always attempt to write** to the **number of nodes** specified in the **replication factor**.

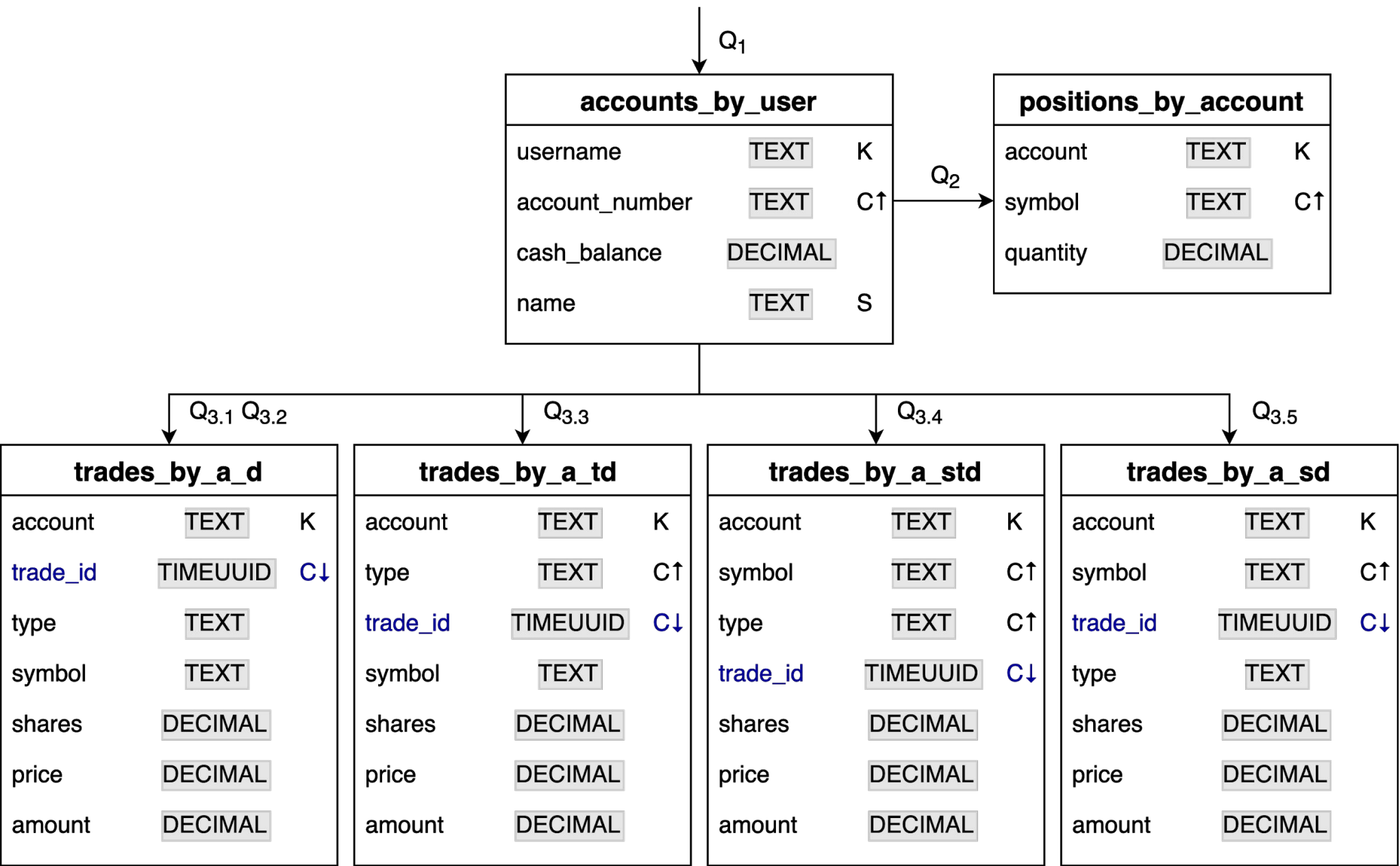
# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. DS210 (Admin) practice
5. **DS220 (Data Modeling) practice**
6. Resources



# 19. Data Modeling (DS220)

Consider the Chebotko Diagram that captures the physical data model for investment portfolio data:



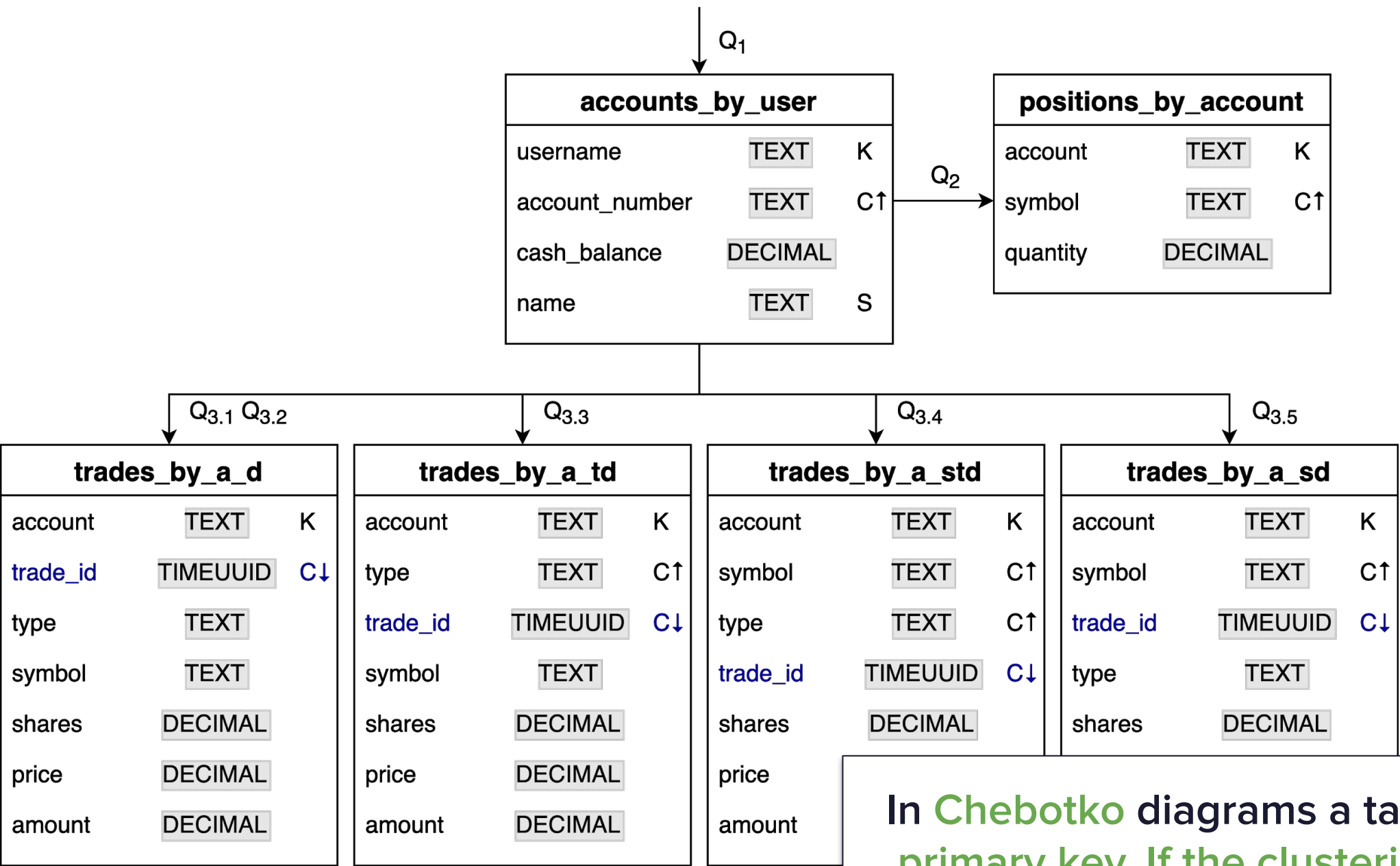
Chebotko Diagram

What is the primary key and clustering order of the table `trades_by_a_sd`?

- A. PRIMARY KEY((account), trade\_id, symbol)  
)  
WITH CLUSTERING ORDER BY (trade\_id DESC, symbol ASC);
- B. PRIMARY KEY((account), trade\_id, symbol)  
)  
WITH CLUSTERING ORDER BY (trade\_id DESC);
- C. PRIMARY KEY((account), symbol, trade\_id)  
)  
WITH CLUSTERING ORDER BY (trade\_id DESC);
- D. PRIMARY KEY((account), symbol, trade\_id)  
)  
WITH CLUSTERING ORDER BY (symbol ASC, trade\_id DESC);

# 19. Solution

Consider the Chebotko Diagram that captures the physical data model for investment portfolio data:



Chebotko Diagram

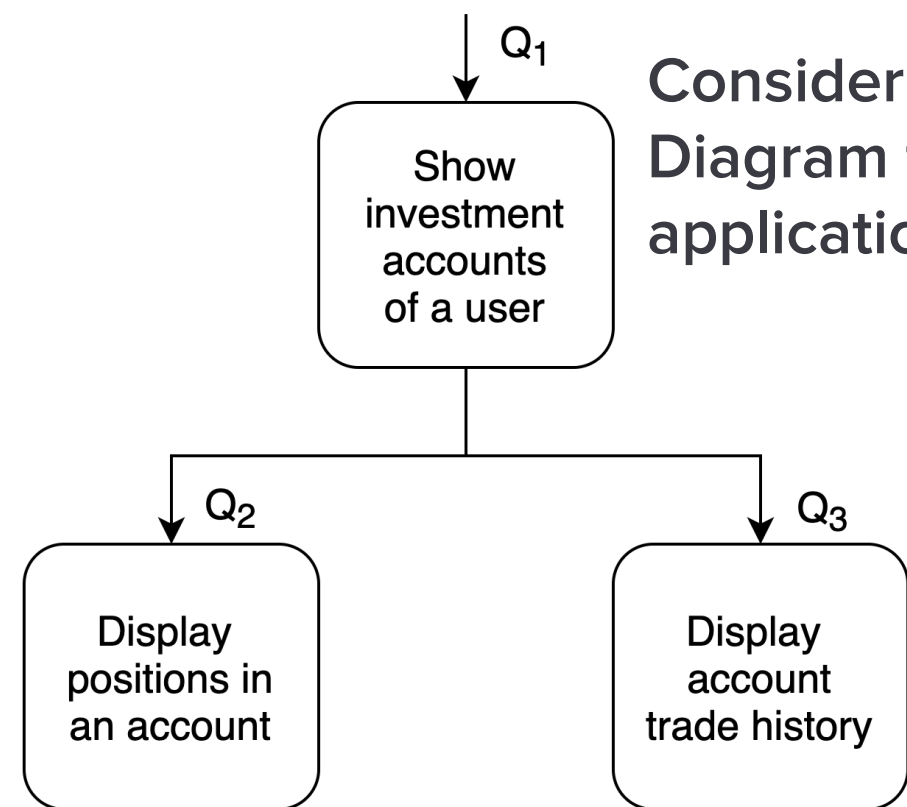
What is the primary key and clustering order of the table trades\_by\_a\_sd?

- A. PRIMARY KEY((account), trade\_id, symbol)  
WITH CLUSTERING ORDER BY (trade\_id DESC, symbol ASC);
- B. PRIMARY KEY((account), trade\_id, symbol)  
WITH CLUSTERING ORDER BY (trade\_id DESC);
- C. PRIMARY KEY((account), symbol, trade\_id)  
WITH CLUSTERING ORDER BY (trade\_id DESC);
- D. PRIMARY KEY((account), symbol, trade\_id)  
WITH CLUSTERING ORDER BY (symbol ASC, trade\_id DESC);

In Chebotko diagrams a table lists clustering keys in the order they appear in the primary key. If the clustering order is explicitly specified for a column with WITH CLUSTERING ORDER BY clause, the clustering order for all preceding clustering key columns must also be explicitly specified.



# 20. Data Modeling (DS220)



Consider the Application Workflow Diagram for an investment portfolio application:

Which access pattern(s) are evaluated before an application can evaluate  $Q_{3.2}$ ?

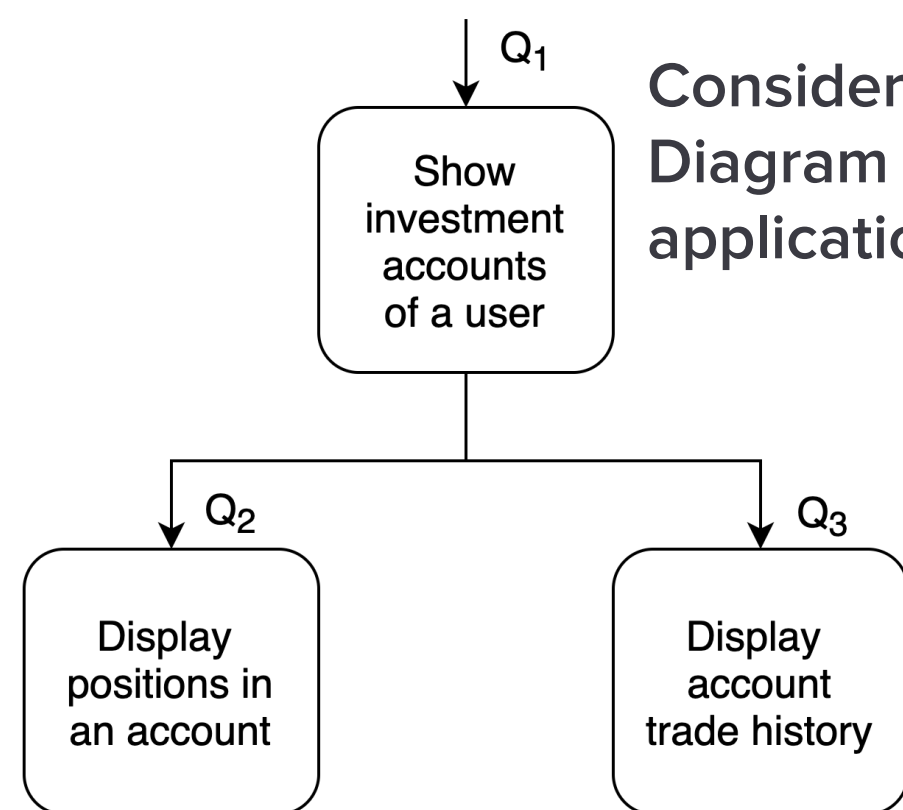
- A.  $Q_1$
- B.  $Q_1$  and  $Q_2$
- C.  $Q_1$  and  $Q_3$
- D.  $Q_1$ ,  $Q_3$  and  $Q_{3.1}$

**Data access patterns**

- $Q_1$ : Find information about all investment accounts of a user
- $Q_2$ : Find all positions in an account; order by instrument symbol (asc)
- $Q_3$ : Find all trades for an account and, optionally, a known date range, transaction type (buy/sell), and stock symbol; order by trade date (desc)
  - $Q_{3.1}$ : Find all trades for an account; order by trade date (desc)
  - $Q_{3.2}$ : Find all trades for an account and date range; order by trade date (desc)
  - $Q_{3.3}$ : Find all trades for an account, date range and transaction type; order by trade date (desc)
  - $Q_{3.4}$ : Find all trades for an account, date range, transaction type and instrument symbol; order by trade date (desc)
  - $Q_{3.5}$ : Find all trades for an account, date range and instrument symbol; order by trade date (desc)



# 20. Data Modeling (DS220)



Consider the Application Workflow Diagram for an investment portfolio application:

Which access pattern(s) are evaluated before an application can evaluate  $Q_{3.2}$ ?

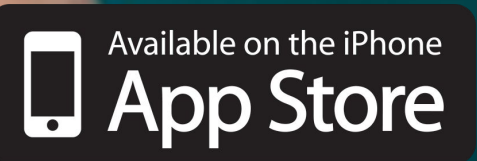
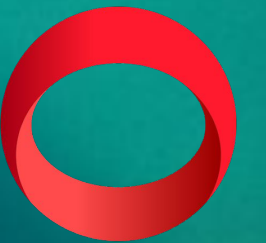
- A.  $Q_1$
- B.  $Q_1$  and  $Q_2$
- C.  $Q_1$  and  $Q_3$
- D.  $Q_1$ ,  $Q_3$  and  $Q_{3.1}$

**Data access patterns**

- $Q_1$ : Find information about all investment accounts of a user
- $Q_2$ : Find all positions in an account; order by instrument symbol (asc)
- $Q_3$ : Find all trades for an account and, optionally, a known date range, transaction type (buy/sell), and stock symbol; order by trade date (desc)
  - $Q_{3.1}$ : Find all trades for an account; order by trade date (desc)
  - $Q_{3.2}$ : Find all trades for an account and date range; order by trade date (desc)
  - $Q_{3.3}$ : Find all trades for an account, date range and transaction type; order by trade date (desc)
  - $Q_{3.4}$ : Find all trades for an account, date range, transaction type and instrument symbol; order by trade date (desc)
  - $Q_{3.5}$ : Find all trades for an account, date range and instrument symbol; order by trade date (desc)

**Q1 is the entry point.** After Q1, Q2 or Q3 may be evaluated. Q3 is broken down into Q3.1 - Q3.5. **The only prerequisite for Q3.1 - Q3.5 is Q1.** Therefore, only Q1 must be evaluated before Q3.2

# menti.com





# Cassandra Certification Workshop

1. Which certification and what resources?
2. Steps for certification
3. DS201 (Foundation) practice
4. DS210 (Admin) practice
5. DS220 (Data Modeling) practice
6. Resources



# MORE LEARNING!!!!

Developer site: [datastax.com/dev](https://datastax.com/dev)

- Developer Stories
- New hands-on learning scenarios with Katacoda
  - Try it Out
  - Cassandra Fundamentals
  - <https://katacoda.com/datastax/courses/cassandra-intro>
  - New Data Modeling course  
<https://katacoda.com/datastax/courses/cassandra-data-modeling>

Classic courses available at [DataStax Academy](https://datastax.com/academy)



**Kata**<oda

# Developer Resources

## LEARN

New hands-on learning at [www.datastax.com/dev](http://www.datastax.com/dev)

Classic courses available at DataStax Academy

## ASK/SHARE

Join [community.datastax.com](http://community.datastax.com)

Ask/answer community user questions – share your expertise

## CONNECT

Follow us [@DataStaxDevs](https://twitter.com/DataStaxDevs)

We are on Youtube – Twitter – Twitch!

## MATERIALS

Slides and practice questions for this course are available at <https://github.com/DataStax-Academy/workshop-cassandra-certificationcassandra-workshop-series>

# Thank You



**DataStax Developers**