KillrChat Presentation

DuyHai DOAN, Technical Advocate

KillrChat presentation

DATASTAX

What is KillrChat ?

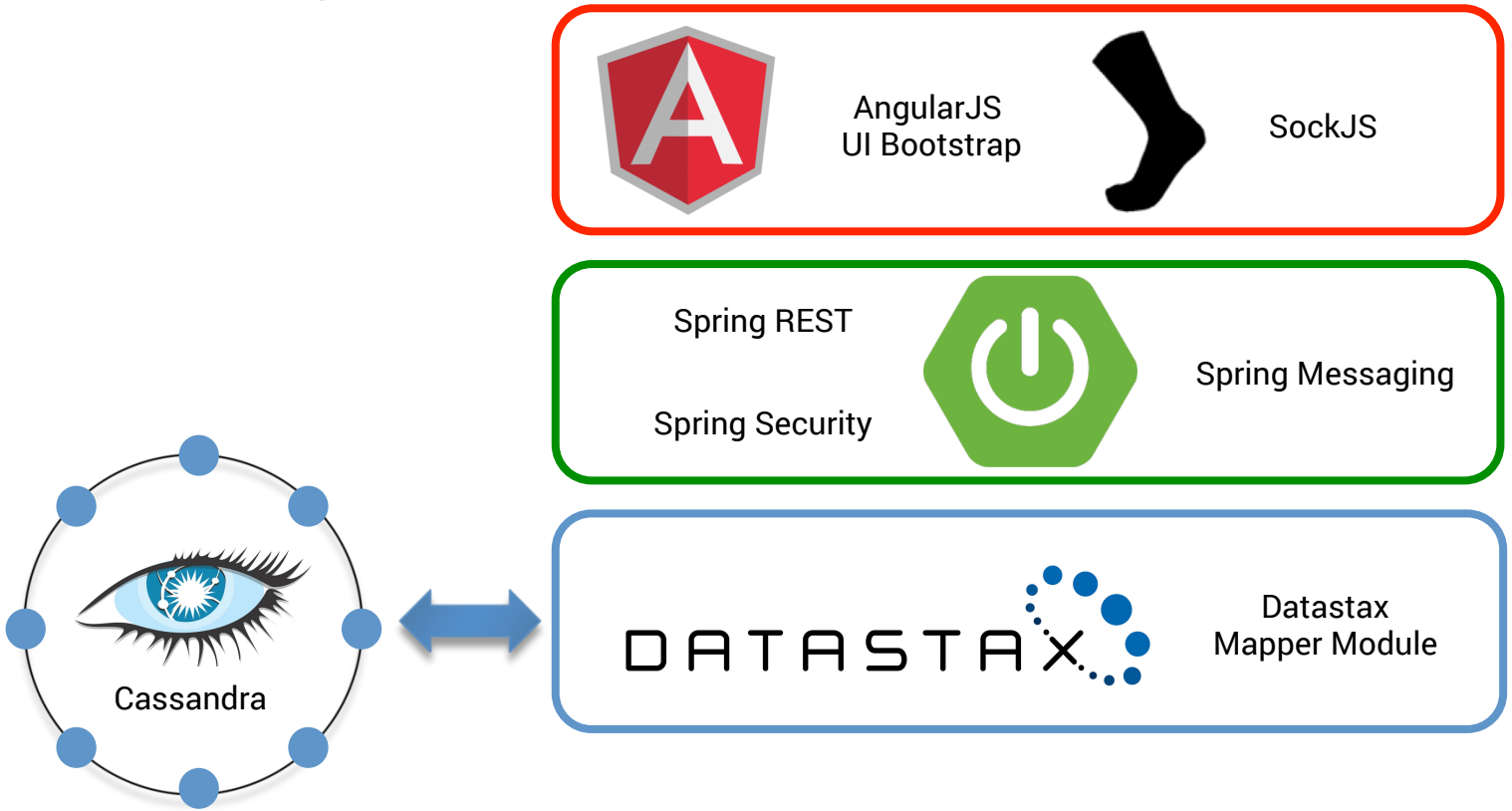
- scalable messaging app

Why KillrChat ?

- show real life de-normalization
- provide real application for attendees
- highlight Cassandra eco-system

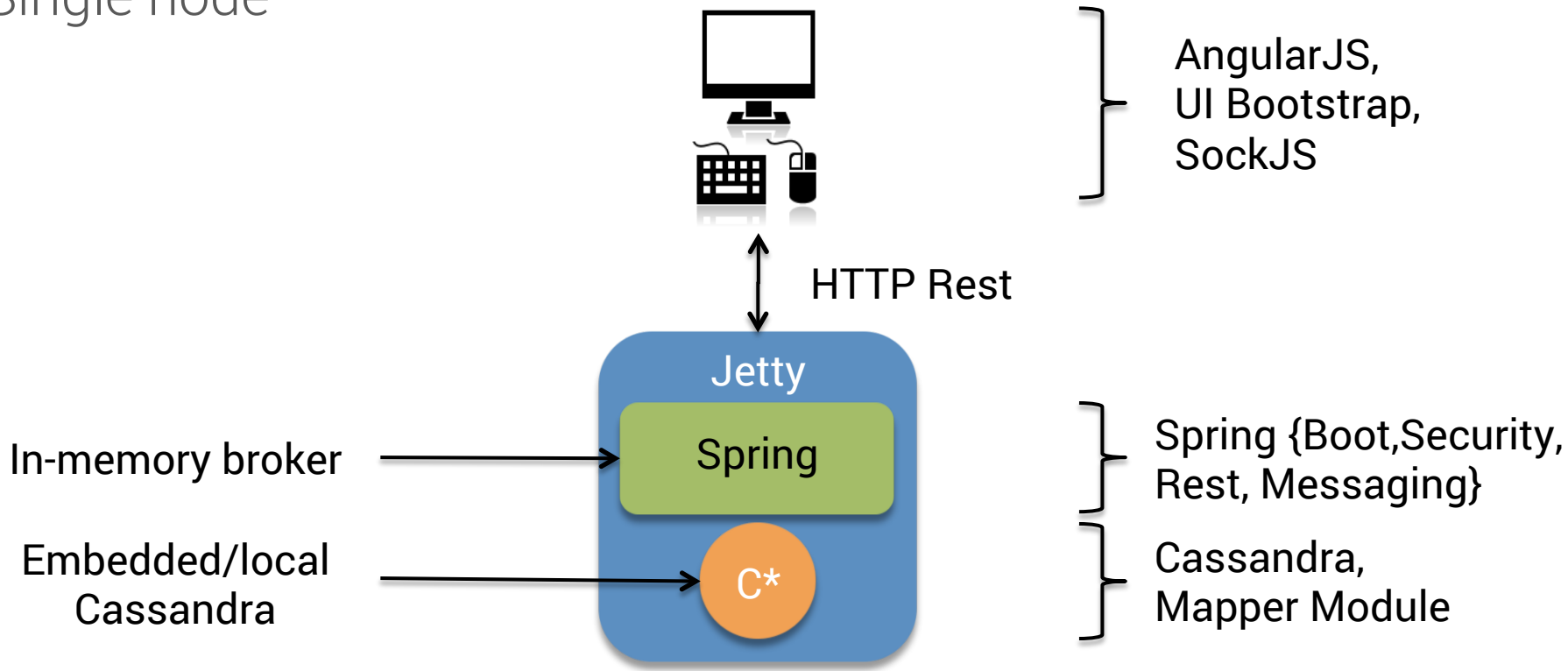
Technology stack

DATASTAX



Architecture

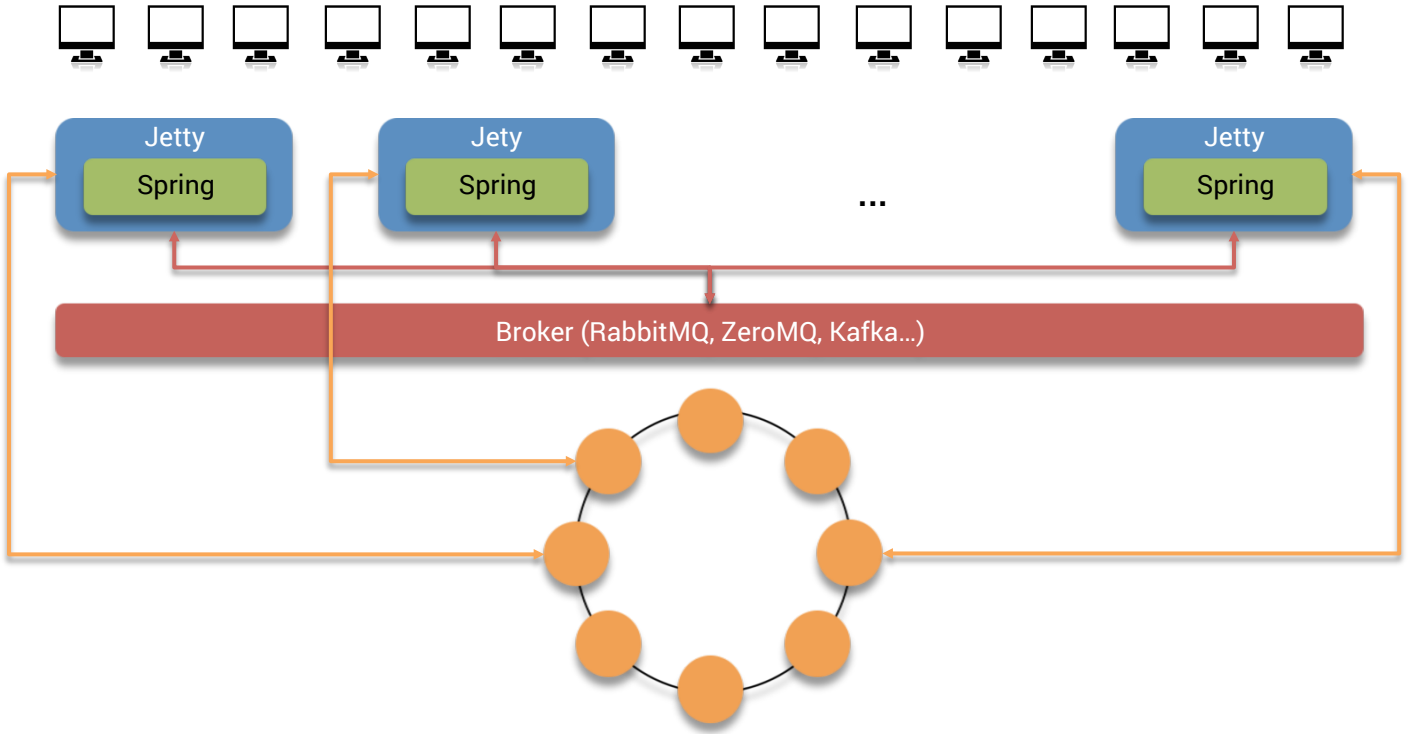
Single node



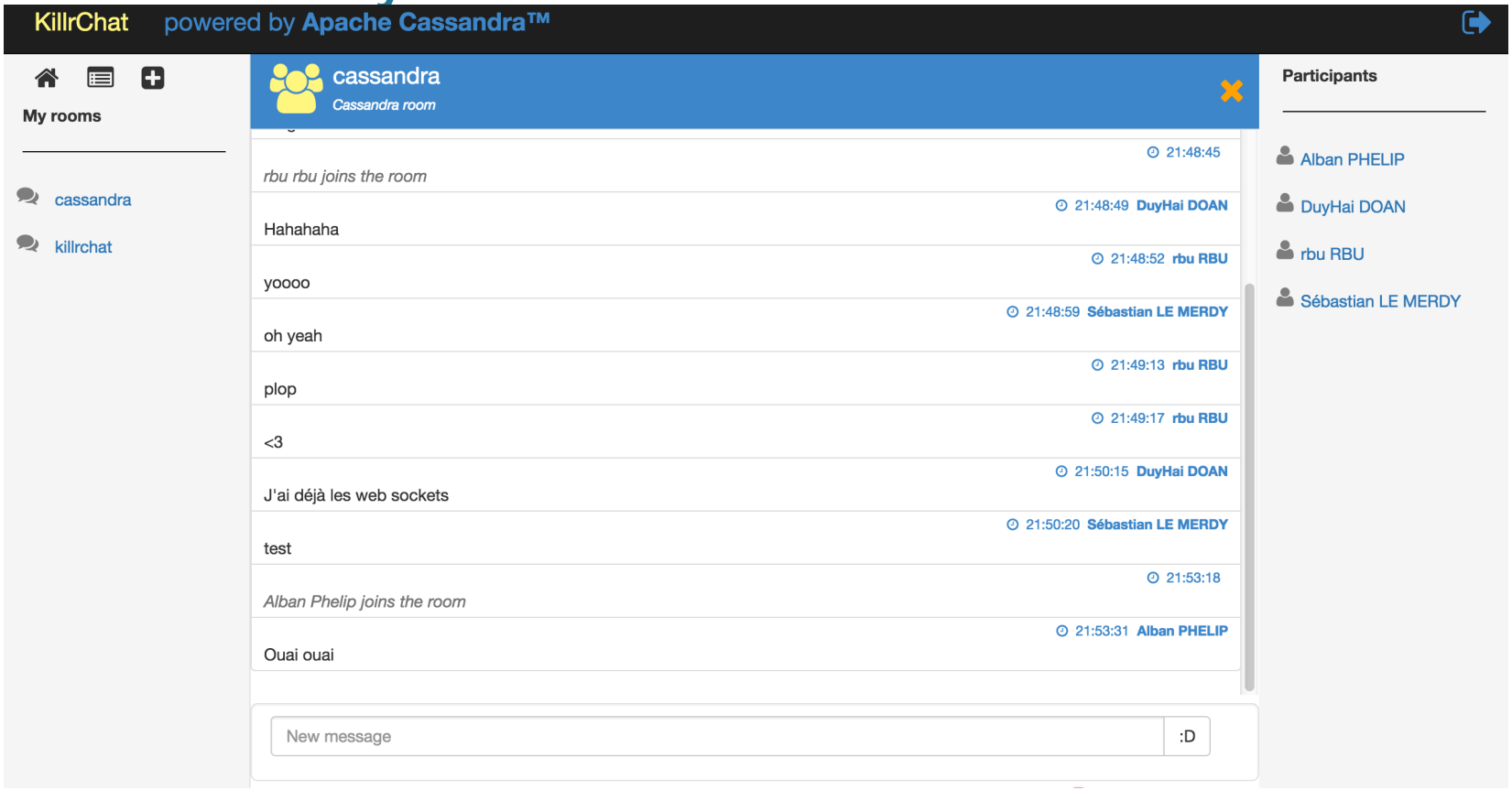
Architecture

DATASTAX

Scaling out



Front end layout



Where can I have it ?

DATASTAX

Clone the Git repository

```
git clone –b CassandraDayParis https://github.com/doanduyhai/killrchat.git
```

Go into the *'killrchat'* folder and launch tests

```
cd killrchat
mvn clean test
```

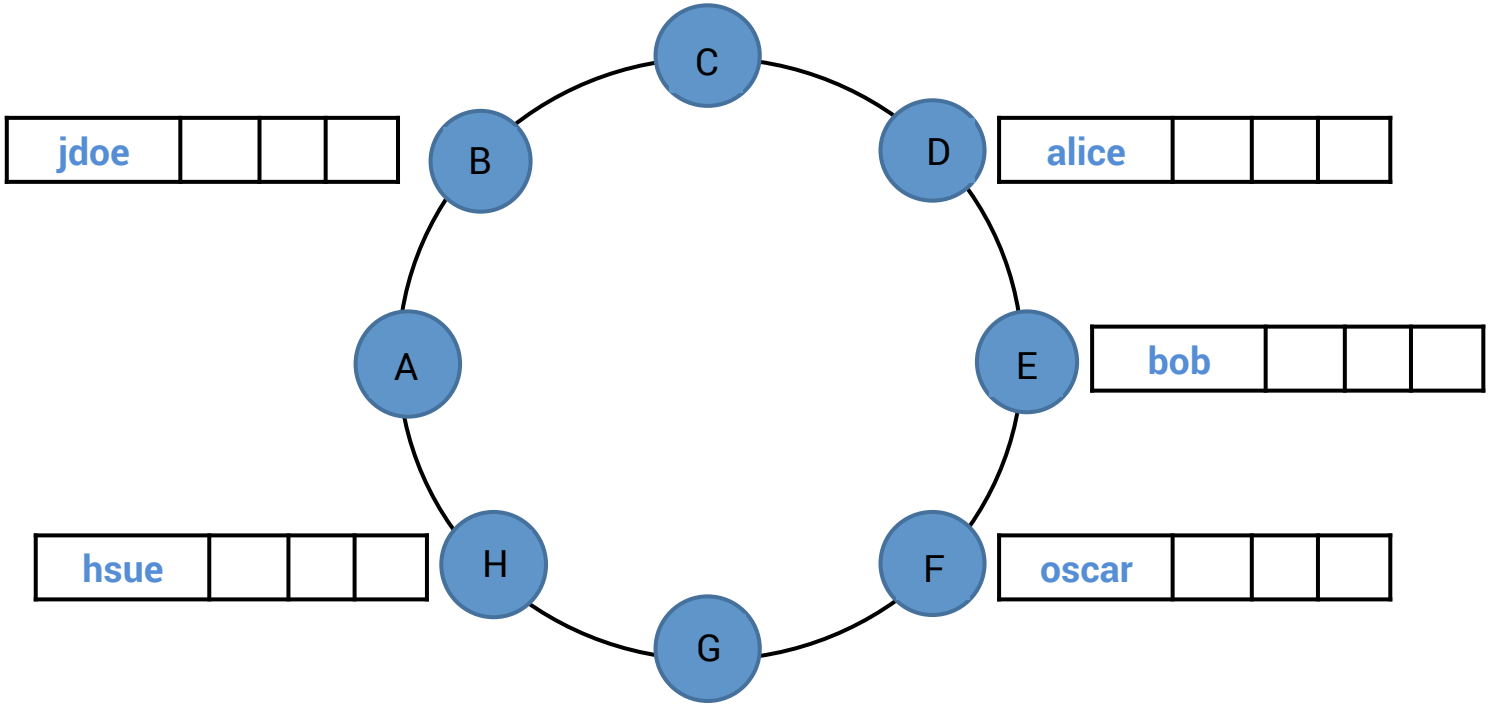
DATASTAX

User Account Management

Scalability

DATASTAX

Scaling by login



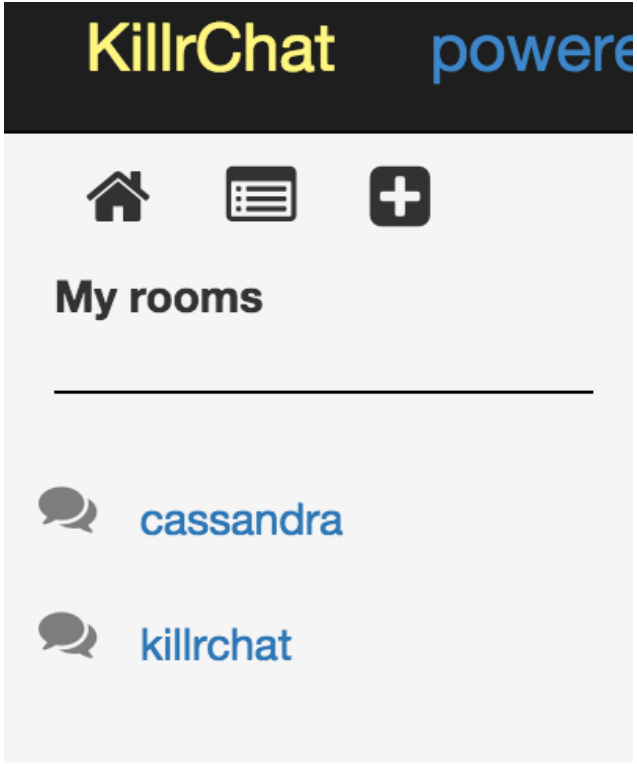
Data model

DATASTAX

```
CREATE TABLE killrchat.users(
        login text,
        pass text,  //password is not allowed because reserved word
        lastname text,
        firstname text,
        bio text,
        email text,
        chat_rooms set<text>,
PRIMARY KEY(login));
```

User’s chat rooms

DATASTAX



User's chat rooms data model

DATASTAX

How to store chat rooms for an user ?

```
CREATE TABLE killrchat.user_rooms(
        login text,
        room_name text,
PRIMARY KEY((login), room_name));
```

- pros: can store huge room count per user (10^6)
- cons: separated table, needs 1 extra SELECT

User's chat rooms data model

DATASTAX

Best choice

```
CREATE TABLE killrchat.users(
    login text,
    ...
    chat_rooms set<text>, //list of chat rooms for this user
PRIMARY KEY(login));
```

- 1 SELECT fetches all data for a given user
- usually, 1 user is not in more that **1000** rooms at a time
- stores only room name

Lightweight Transaction

DATASTAX

Avoid creating the same login by 2 different users ?

☞ use Lightweight Transaction

```
INSERT INTO killrchat.users(room_name, …)
VALUES (‘jdoe’, …) IF NOT EXISTS ;
```

Expensive operation

☞ do you create a new account every day ?

Demo

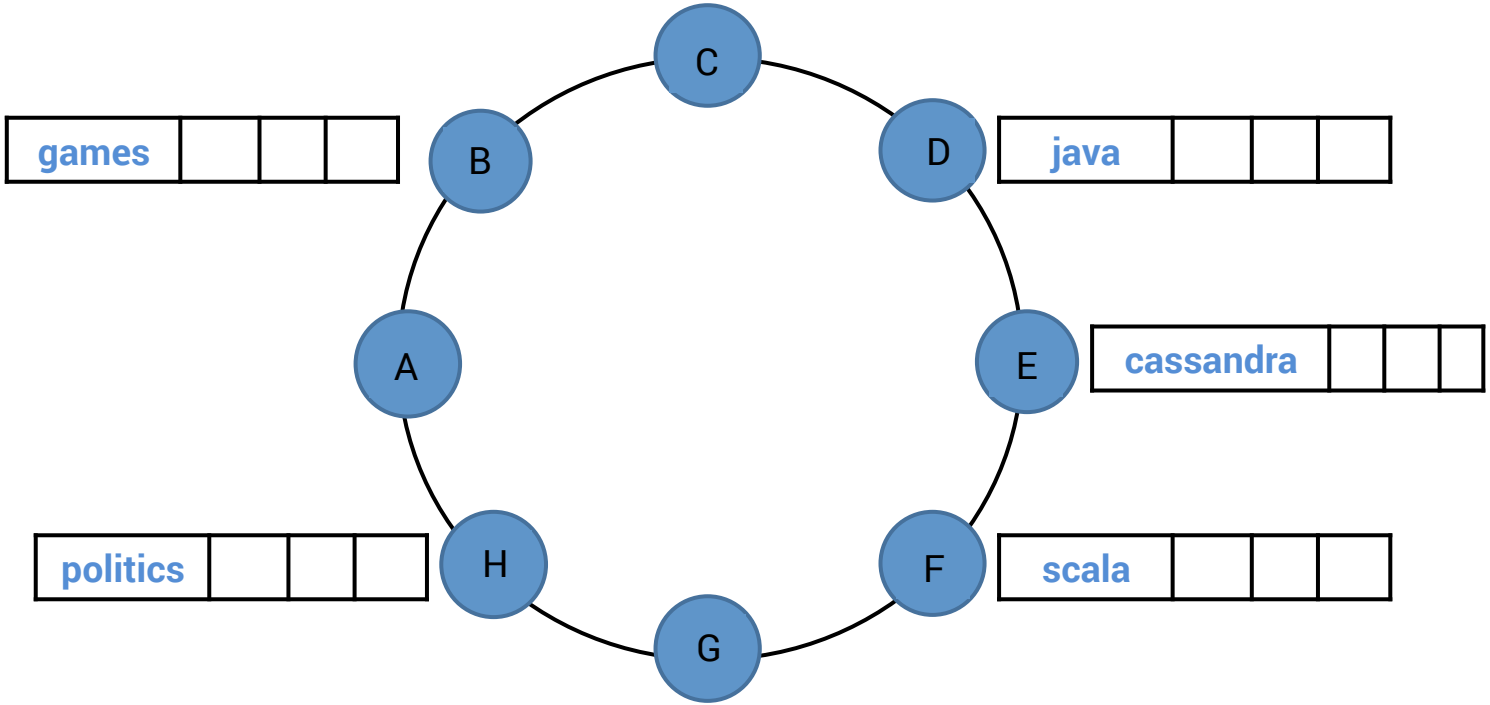
DATASTAX

Chat Room Data Model

Scalability

DATASTAX

Scaling by room name

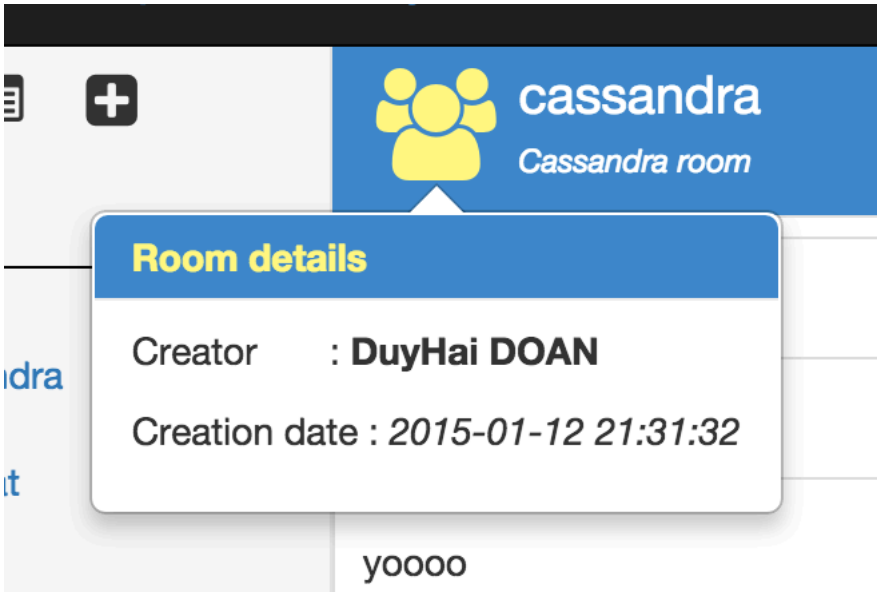


Data model

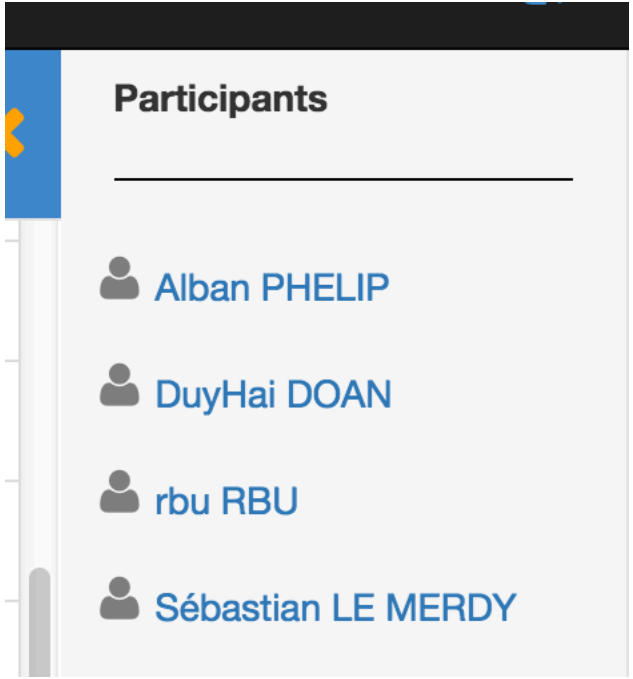
DATASTAX

```
CREATE TABLE killrchat.chat_rooms(
        room_name text,
        creation_date timestamp,
        banner text,
        creator ???,
        creator_login text,
        participants ???,
PRIMARY KEY(room_name));
```

Room details



Room participants



De-normalization

DATASTAX

```
CREATE TYPE killrchat.user(
        login text,
        firstname text,
        lastname text);

CREATE TABLE killrchat.chat_rooms(
        ...
        creator frozen<user>,                   // de-normalization
        ...
        participants set<frozen<user>>,         // de-normalization
PRIMARY KEY(room_name));
```

Lightweight Transaction

DATASTAX

Avoid creating the same room by 2 different users ?

☞ use Lightweight Transaction

```
INSERT INTO killrchat.chat_rooms(room_name, …)
VALUES (‘games’, …) IF NOT EXISTS ;
```

Demo

DATASTAX

Chat Room Management

Participants management

Room deletion

Participant joining

DATASTAX

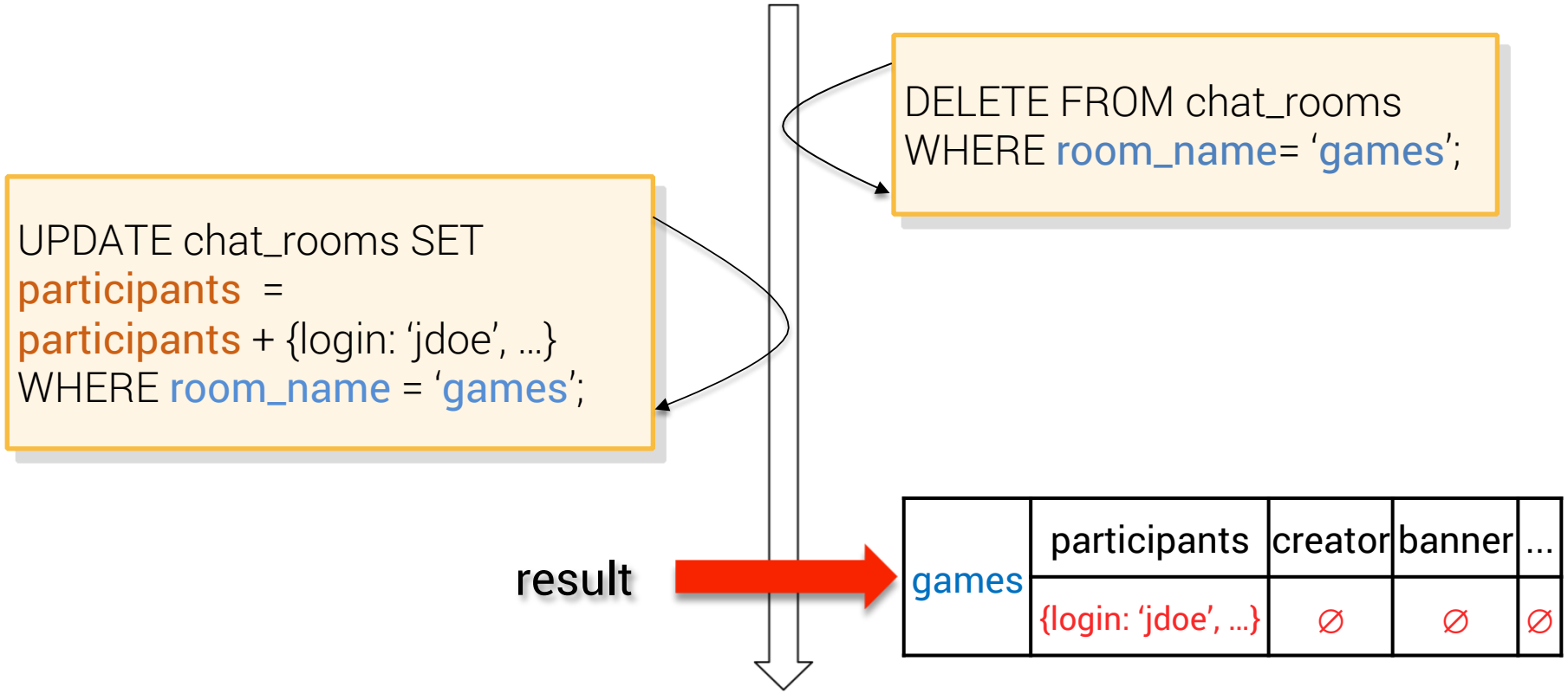
Adding new participant

```
UPDATE killrchat.chat_rooms SET participants  = participants + {...}
WHERE room_name = 'games';
```

What if the creator deletes the room at the same time ?

Concurrent delete/update

DATASTAX



Participant joining

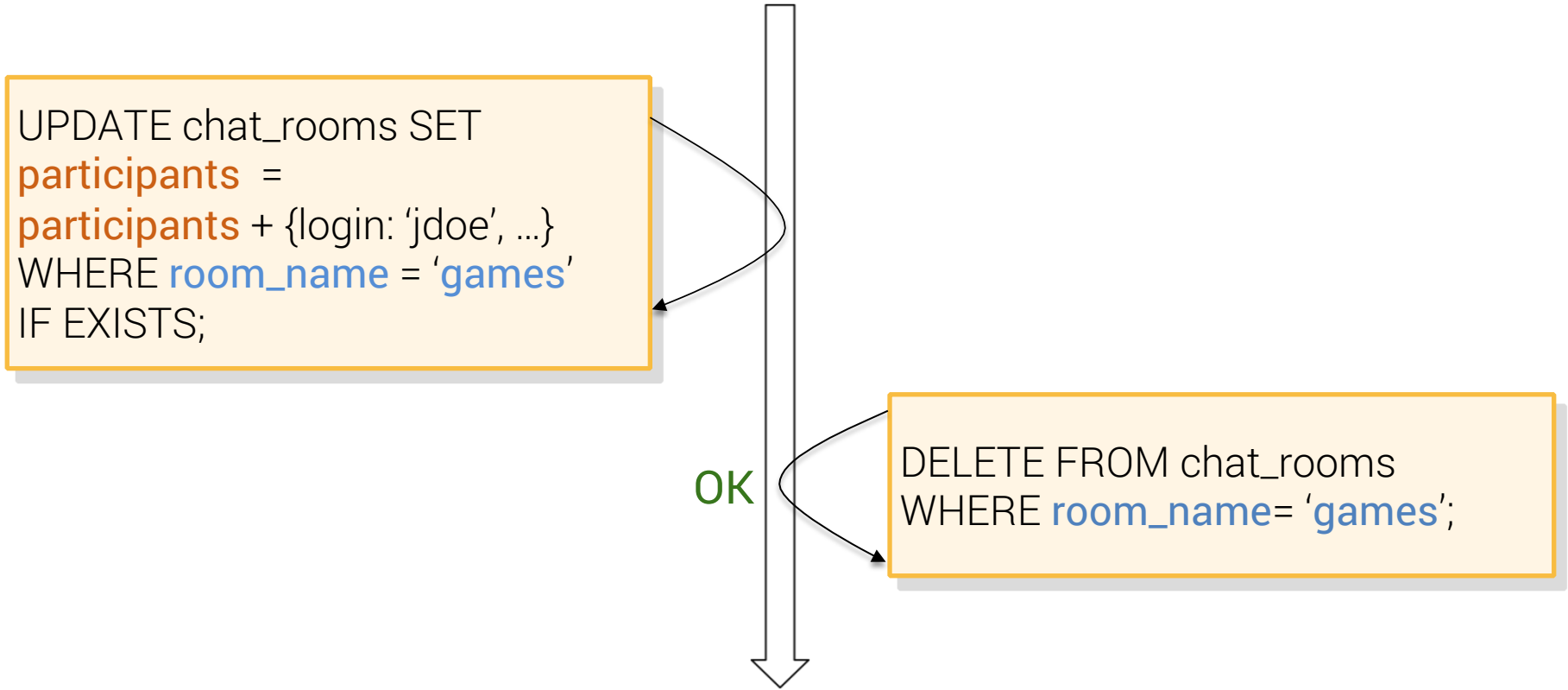
DATASTAX

Solution

☞ use Lightweight Transaction

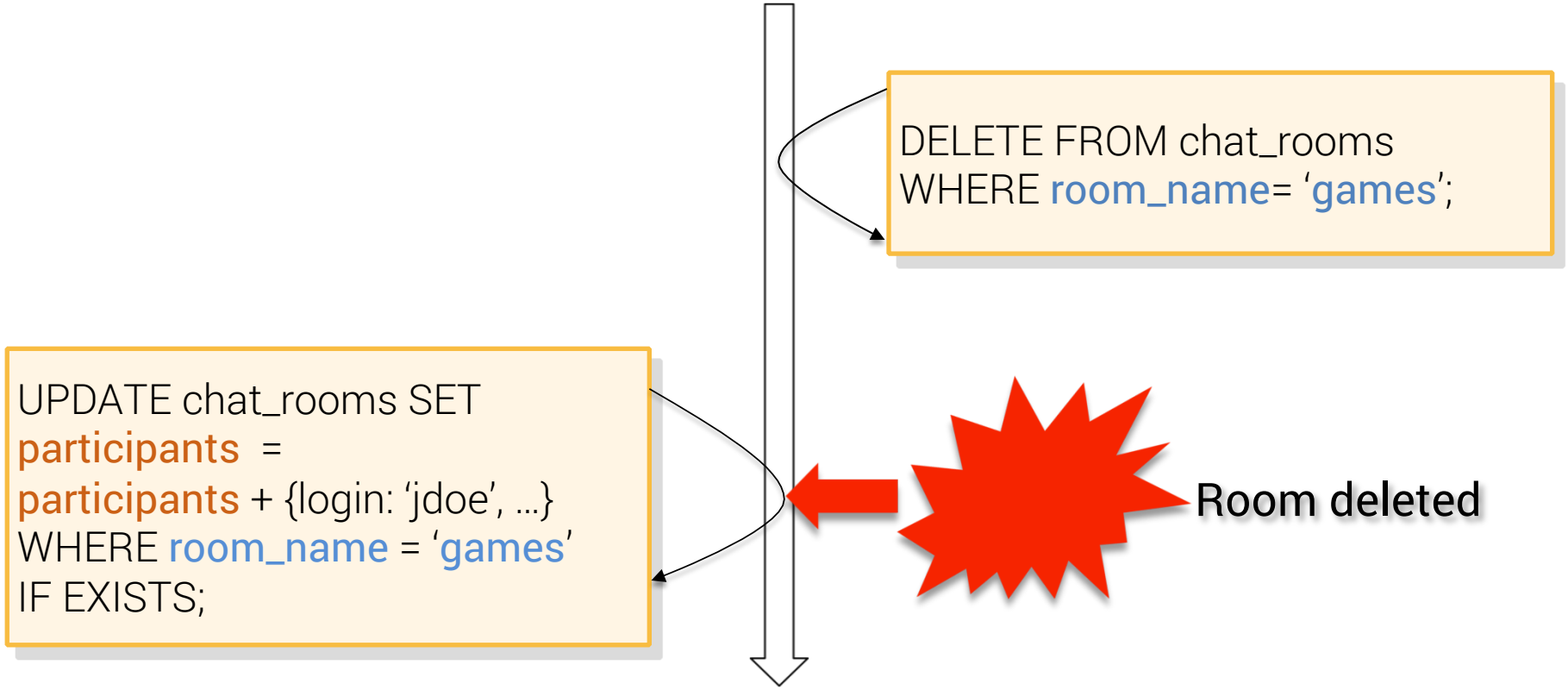
```
UPDATE killrchat.chat_rooms SET participants  = participants + {...}
WHERE room_name = 'games' IF EXISTS;
```

Concurrent delete/update



Concurrent delete/update

DATASTAX



Participant leaving

DATASTAX

Removing participant (no read-before-write)

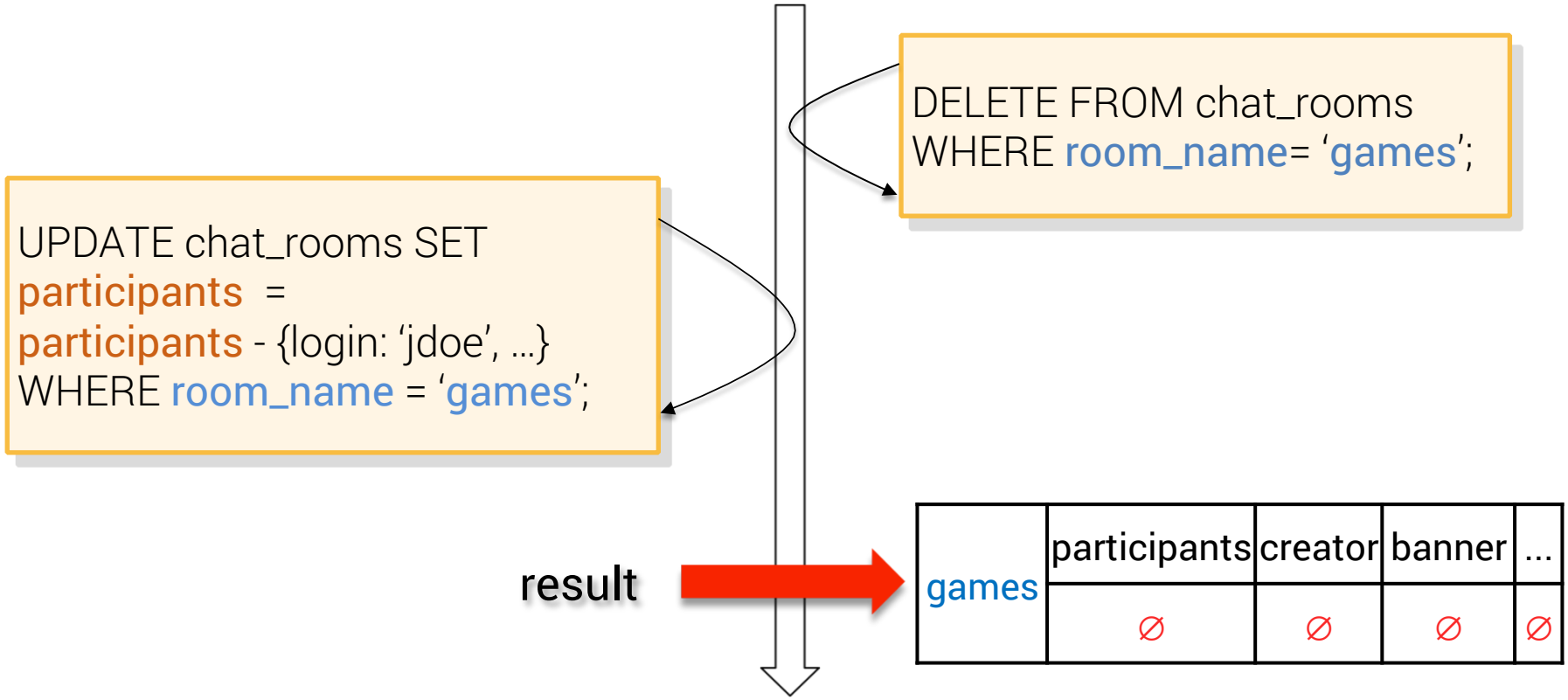
```
UPDATE killrchat.chat_rooms SET participants  = participants - {…}
WHERE room_name = 'games';
```

What if the creator deletes the room at the same time ?

- we'll create a tombstone
- tombstone will be garbage-collected by compaction

Concurrent delete/update

DATASTAX



Deleting room

DATASTAX

What if participant leaving at the same time ?

- not a problem, tombstone will be garbage

What if participant joining at the same time ?

☞ use Lightweight Transaction

Only room creator can delete room, no one else!

☞ use Lightweight Transaction

Deleting room

DATASTAX

Solution

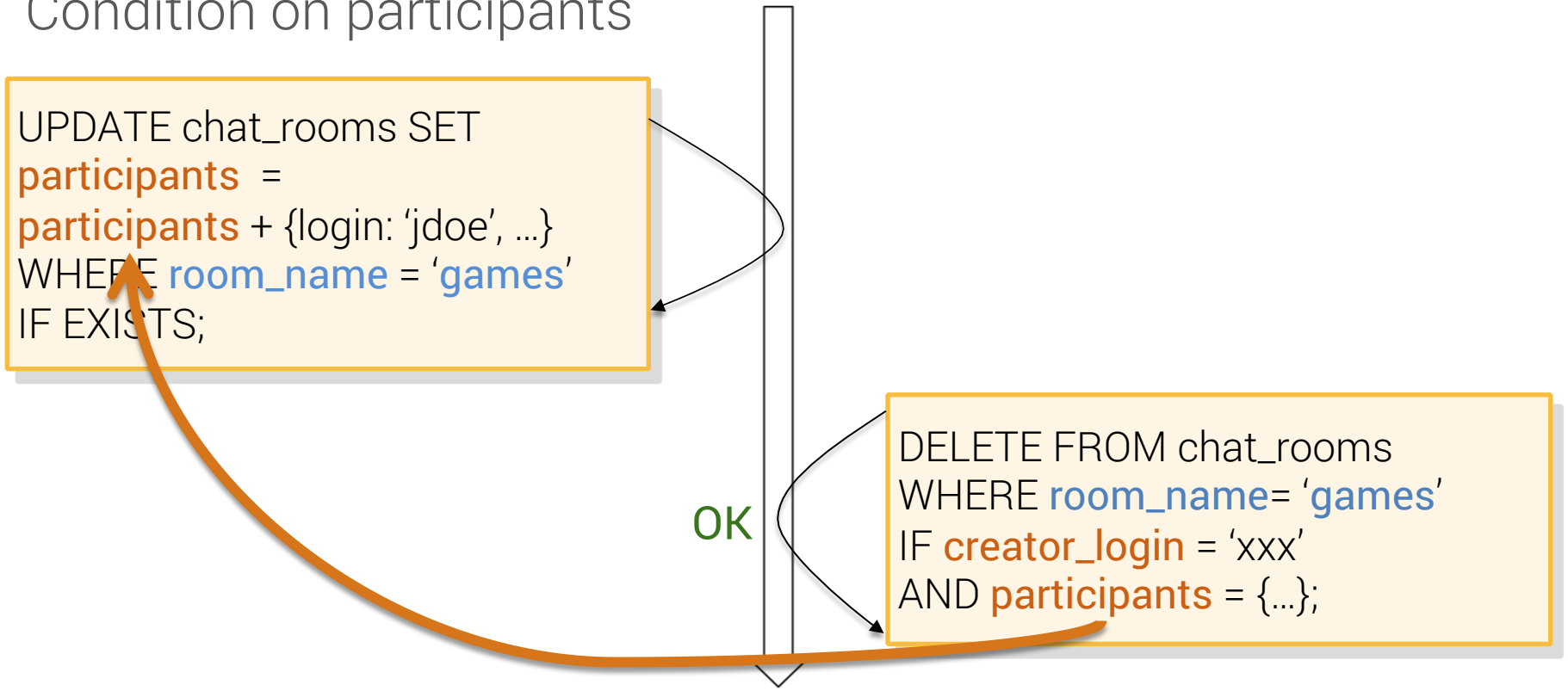
```
DELETE killrchat.chat_rooms
WHERE room_name = 'games'
IF creator_login = <current_user_login> AND participants = {...};
```

Condition on creator_login

- current user login coming from **security context**, no cheating !

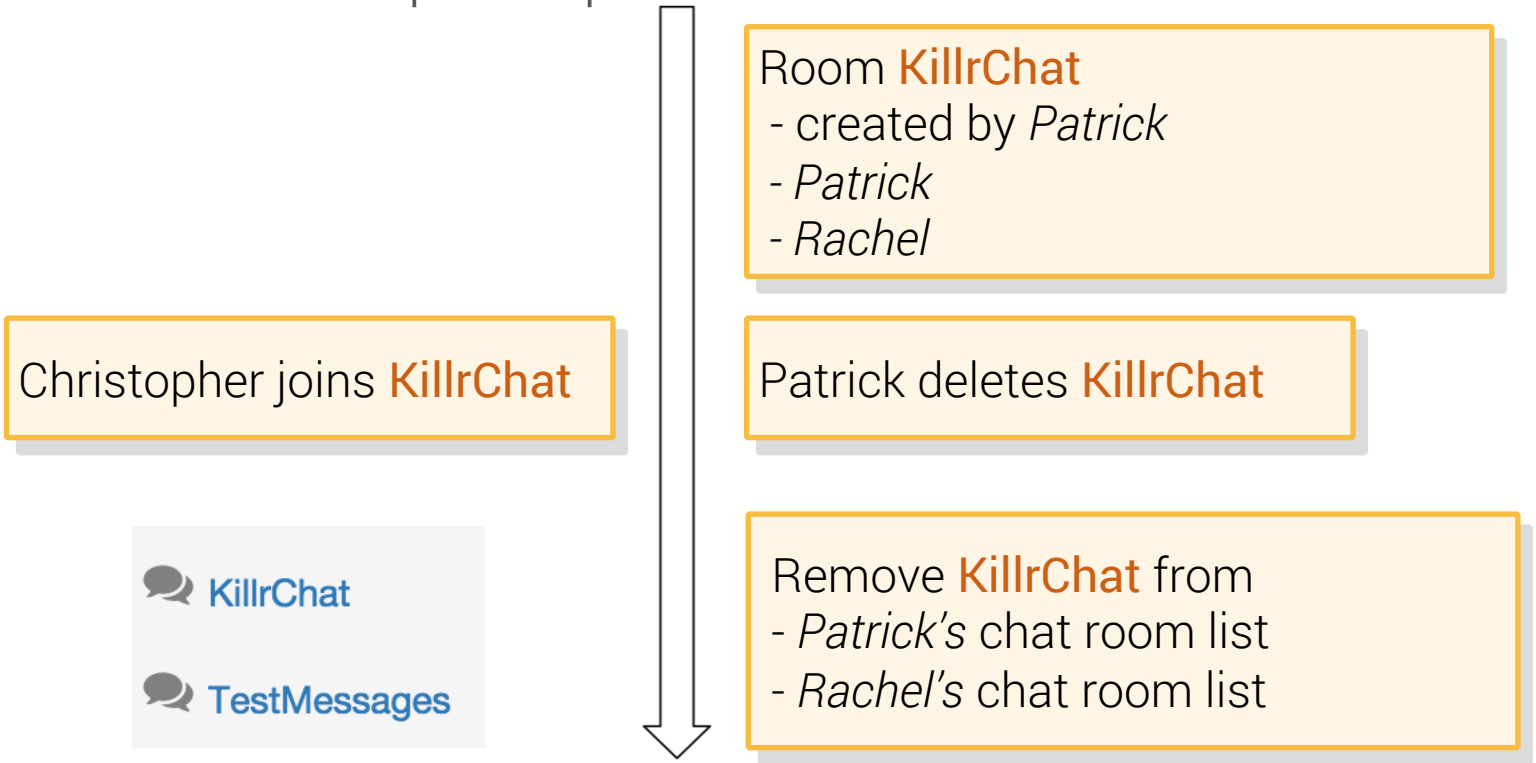
Deleting room

Condition on participants



Deleting room

Without condition on participants



Deleting room

DATASTAX

LightWeight Transaction is slow ....

... but how often do you delete rooms ?

Demo

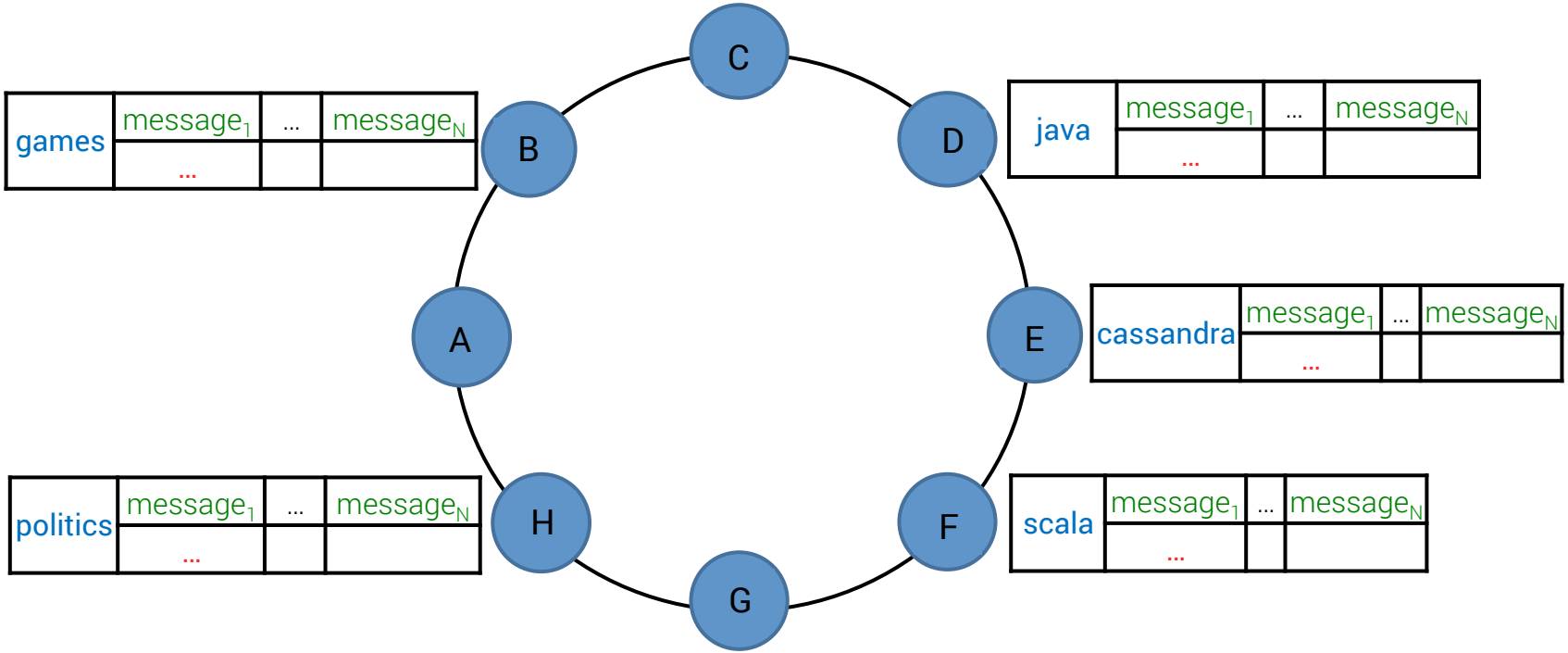
DATASTAX

Chat Messages

Scalability

DATASTAX

Scaling messages by room name



Data model

```
CREATE TABLE killrchat.chat_room_messages(
        room_name text,
        message_id timeuuid,
        content text,
        author frozen<user>,                    // denormalization
        system_message boolean,
        PRIMARY KEY((room_name), message_id)
) WITH CLUSTERING ORDER BY (message_id DESC);
```

Data model

DATASTAX

Clustering column **message_id** order by DESC

- latest messages first

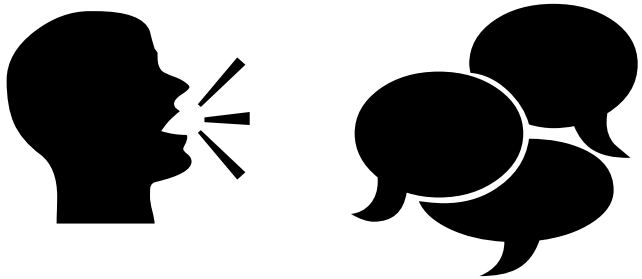
Improvements

- current data model limits messages count to ≈ 500 x 10^6
- bucketing by day is the right design

PRIMARY KEY((room_name, day), message_id) *//day format **yyyyMMdd***

Demo

DATASTAX



Q & R

Thank You

@doanduyhai

duy_hai.doan@datastax.com

https://academy.datastax.com/