Scott Viet Phong Nguyen, 350313

Anh Duc Nguyen, 379744

Task 1 – Cassandra Query Language (5+10+5+5+5+10+10 = 50%)

a) Keyspaces

Does the following CQL3 statement create a new keyspace that places one replica in each of two data centers?

cqlsh> CREATE KEYSPACE <ksname>

WITH REPLICATION = {'class':'NetworkTopologyStrategy','DC1':1,'DC2':1};

Yes

_____

b) Compound primary keys

Which statement creates a table "messages" in the keyspace "msgplatform" that is partitioned by "participant_id" across multiple servers and locally clustered by "posted_on" date and "posted_location" string?

```
CREATE TABLE msgplatform.message(
   participant_id int,
   posted_on timestamp,
   posted_location text,
   PRIMARY KEY (participant_id,(posted_on, posted_location))
)
```

_____

c) Add a collection to a table

You want to alter an existing "customers" table schema with an additional column that can contain sets of "interests" (text strings). After you have altered the table, a query like this should work:

cqlsh:msgplatform> UPDATE customers SET interests = interests + {'climbing','baking'} WHERE id = 202;

Please complete the following uncomplete statement using correct SQL syntax:

```
cqlsh:msgplatform> ALTER TABLE customers ADD interests set<text>;
```

_____

d) TTL

Is it true or false? The following statement inserts an expiring column named "pw_reset_token" with value "fgh-lmn-456" into the "customers" table. The column "pw_reset_token" will be deleted automatically after 6000 seconds but the customer row with id 315 and name "karlmann" will still be present in the table after 6000 seconds.

cqlsh:msgplatform> INSERT INTO customers (id, name,

pw_reset_token)

VALUES (315, 'karlmann', 'fgh-lmn-456')

USING TTL 12000;

False

_____

e) Counter columns

You have created a table named "referendum" with two counter columns like this:

cqlsh:msgplatform> CREATE TABLE referendum (

participant_id int,

msg_created_time timeuuid,

msg_created_location string,

voteinfavor counter,

voteoppose counter,

PRIMARY KEY (participant_id, msg_created_time,)

);

Complete the following query that increments the number of in favor votes for a referendum which are created in "germany", between the first and second of January 2016 (midnight-midnight) and from the participant with the number "519":

cqlsh:msgplatform> UPDATE referendum SET voteinfavor = voteinfavor + 1 WHERE participant_id = 519

AND msg_created_time >= maxTimeuuid(2016-01-01 00:00:00+0000)

AND msg_created_time <= minTimeuuid(2016-01-02 00:00:00+0000)

AND msg_created_location = 'germany';

_____

f) Lightweight transactions with Compare-and-Set (CAS)

Write two CQL queries that implement the following feature:

1. A user named Catarina (user id: “689”) wants to reset her password by creating an expiring token ("pw_reset_token") column in her user row in the “customers” table.

2. She gets a message with a link to a site where to reset her password. On that site she can enter a new password only if the expiring token (encoded as parameter into the link in her message) matches the expiring token in her user row and if the token has not expired, yet.

```
UPDATE customers USING TTL 1800
SET pw_reset_token = 'pw_token'
WHERE userid = 689
```

```
UPDATE customers
SET password = 'newpassword'
WHERE userid = 689
IF pw_reset_token = 'some-generated-reset-token'
```

_____

g) True or false?

Only one node in a P2P data cluster (such as a Cassandra cluster) maintains a sliding window of inter-arrival times of Gossip messages.

False

Task 2 – MongoDB – Modeling a Scenario (50%)

We assume that the first line creates an Object with the _id: ObjectId("123").

We also filled <YYYY-mm-dd> with an actual date.

1. db.testing.save({startDate:new Date()});
2. db.testing.update({"_id":ObjectId("123")},{"$set":{"previousMeterNumber": 123456789,"installments":[]}});
3. db.testing.update({"_id":ObjectId("123")},{"$set":{"meter": []}});
4. db.testing.update({"_id":ObjectId("123")},{"$set":{"customer": []}});
5. db.testing.update({"_id":ObjectId("123")},{"$push":{"installments": {startDate: new Date(), grossValue:39}}});
6. db.testing.update({"_id":ObjectId("123")},{"$push":{"installments": {startDate: new Date(), grossValue:40}}});
7. db.testing.update({"_id":ObjectId("123")},{"$push":{"meter": {readingDate: new Date(), meterValue:20}}});
8. db.testing.update({"_id":ObjectId("123")},{"$push":{"meter": {readingDate: new Date(), meterValue:40}}});
9. db.testing.update({"_id":ObjectId("123")},{"$push":{"meter": {readingDate: new Date(), meterValue:60}}});
10. db.testing.update({"_id":ObjectId("123")},{"$set": {"customer": {company: "Smith Inc.",email: "john@smith.com"}}});
11. db.testing.update({„_id“:ObjectId(„123“)},{„$set“: {"customer.firstName": "John","customer.lastName": "Smith","customer.address":[]}});
12. db.testing.update({"_id":ObjectId("123")},{"$push":{"customer.address": {street: "UpperStreet", streetNumber:15, zip:10411,"city":"Berlin"}}});
13. db.testing.update({"_id":ObjectId("123")},{"$push":{"customer.address": {street: "LowerStreet", streetNumber:19, zip:10413,"city":"Berlin"}}});
14. db.testing.update({"_id":ObjectId("123")},{"$set": {"customer.bankAccounts": []}});
15. db.testing.update({"_id":ObjectId("123")},{"$push":{"customer.bankAccounts": {IBAN: 123456789, BIC:112233445, "bankName":"BankofEngland"}}});
16. db.testing.update({"_id":ObjectId("123")},{"$push":{"customer.bankAccounts": {IBAN: 123456799, BIC:552233445, "bankName":"BankofScottland"}}});
17. db.testing.update({"_id":ObjectId("123")},{"$set": {"tariff": {netBasePrice: 4.99,VAT:19,EEGLevy:6.354}}});