Professor: Louridas Panos  
Student: Chatzimoschou Angeliki

BAPT XXXX  
July 2016

Big data systems  
cassandra assignment

**E-R Diagram**

****

**Relational Database Schema:**

Tha main and most important characteristic of Cassandra database, which differentiates her at most from relational databases is denormalization. As there are no Joins, we start with the queries and organize the data around them.   
Therefore, we design our tables so that we minimize partitions that must be searched to   
answer a query.



**Music\_Keyspace**

We create our keyspace: MUSIC\_keyspace set it for use.

CREATE KEYSPACE music\_keyspace WITH replication = {'class':'SimpleStrategy', 'replication\_factor': '1'}  
AND durable\_writes = true;

use music\_keyspace;

**Q1: Find user by name**

**Create ‘address’ and ‘user’ tables**

We will create a new type of data: address, in order to store separately and at once all values of a user's address. Then we will create ‘user’ table. In order to save space we don’t set any ID column, since we can use each user’s name (which is unique for each user) as Primary Key.

CREATE TYPE address (  
street text,  
city text,  
postal\_code int,  
country text);

CREATE TABLE user(  
name text,  
paymentInfo text,  
address map<text,frozen<address>>,  
PRIMARY KEY(name)  
)WITH COMMENT ='Q1 Find user by name';

**Insert dummy data to ‘user’ table**

INSERT INTO user(name,paymentInfo,address)  
VALUES ('AngelicaChatz','PayPal',{'home':{street:'Mistra 12' ,city:'Athens',postal\_code:16561,country:'Greece'}});

INSERT INTO user(name,paymentInfo,address)  
VALUES ('AnastasiaRilmaheri','credit card',{'home':{street:'Platonos 120', city:'Athens',postal\_code:11253,country:'Greece'}});

**Query**

SELECT \* FROM user WHERE name='AnastasiaRilmaheri';

**Q2: Find song by name**

**Create ‘song’ table**

CREATE TABLE song(  
song\_name text,  
artist text,  
album text,  
genre text,  
year int,  
count int,  
song\_file blob,  
PRIMARY KEY(song\_name)  
)WITH COMMENT='Q2 Find song by name';  
  
**Insert dummy data to ‘song’ table**

INSERT INTO song(song\_name,artist,album,genre,year,count)  
VALUES ('Sexual Revolution','Army of Lovers','Glory Glamour and Gold',  
'Dance',1994,6850045);

**Query**SELECT \* FROM song WHERE song\_name='Sexual Revolution';

**Q3: Find songs played by a user, arranged in reversed chronological order (more recent first)**

We will set as primary key all attributes, so as to keep only one record if a user has listened to one song during the same day. In addition, we will use clustering order, by which the results of the query will be presented in descending order by date played and ascending order by song name.

**Create ‘songs\_by\_user’ table**

CREATE TABLE songs\_by\_user(  
song\_name text,  
user\_name text,  
date\_played date,  
PRIMARY KEY (user\_name,date\_played,song\_name)  
)WITH CLUSTERING ORDER BY (date\_played DESC,song\_name ASC)  
AND COMMENT='Q3:Find songs played by a user (reverse chronological order)';

**Insert dummy data to ‘songs\_by\_user’ table**

INSERT INTO songs\_by\_user(song\_name,user\_name,date\_played)  
VALUES ('lost on you','AngelicaChatz','2016-12-25');

INSERT INTO songs\_by\_user(song\_name,user\_name,date\_played)   
VALUES ('partition','AnastasiaRilmaheri','2016-11-13');

INSERT INTO songs\_by\_user(song\_name,user\_name,date\_played)   
VALUES ('cheyenne','AnastasiaRilmaheri','2016-08-19');

INSERT INTO songs\_by\_user(song\_name,user\_name,date\_played)   
VALUES ('la vida es un carnaval','AngelicaChatz','2016-09-06');

**Query**  
  
SELECT \* FROM songs\_by\_user WHERE user\_name='AngelicaChatz' ORDER BY date\_played DESC;

**Q4: Find playlist by name**

**Create ‘playlist’ table**  
CREATE TABLE playlist(  
playlist\_name text,  
description text,  
genre text,  
creator text,  
PRIMARY KEY (playlist\_name)  
)WITH COMMENT ='Q4 Find playlist by name';

**Insert dummy data to ‘playlist’ table**

INSERT INTO playlist (playlist\_name,description, genre, creator)  
VALUES ('Dance Hits','Dance Music','Dance','AngelicaChatz');

**Query**   
  
SELECT \* FROM playlist WHERE playlist\_name='Dance Hits';

**Q5: Find playlist by genre**

Since genre is of medium cardinality, instead of creating a new table, we can use the method of indexing.

**Create Index**  
  
CREATE INDEX ON playlist(genre);

**Query**

SELECT \* FROM playlist WHERE genre='Dance';

**Q6: Find playlist by creator**

We will use materialized view since the creator is of high cardinality and thus indexing is not a good choice.

**Create Materialized View**

CREATE MATERIALIZED VIEW playlist\_by\_creator  
AS SELECT creator, playlist\_name  
FROM playlist  
WHERE creator IS NOT NULL AND playlist\_name IS NOT NULL  
PRIMARY KEY (creator, playlist\_name)  
WITH COMMENT ='Q6:Find playlist by creator';

**Query**  
SELECT \* FROM playlist\_by\_creator WHERE creator = 'AngelicaChatz';

**Q7: Find the followers of a playlist**

We will create a new table since we could have millions of followers of a playlist and we cannot insert more than 64K values. As primary key we set both playlist\_name and follower, since we want to avoid multiple common stores to our table.

**Create ‘playlist\_followers’ table**CREATE TABLE playlist\_followers(  
playlist\_name text,  
follower text,   
PRIMARY KEY(playlist\_name,follower)  
)WITH COMMENT = 'Q7 Find the followers of a playlist';

**Query**

SELECT follower FROM playlist\_followers WHERE playlist\_name='Dance Hits';

**Q8: Find the followers of a user**

**Create ‘user\_followers’ table**  
CREATE TABLE user\_followers(  
user\_name text,   
follower text,   
PRIMARY KEY(user\_name,follower)  
)WITH COMMENT = 'Q8 Find the followers of a user';

**Insert dummy data to ‘user\_followers’ table**INSERT INTO user\_followers(user\_name, follower)  
VALUES ('AngelicaChatz','AnastasiaRilmaheri');

INSERT INTO user\_followers(user\_name, follower)  
VALUES ('AngelicaChatz','ChieIsilwen');

**Query**

SELECT follower FROM user\_followers WHERE user\_name='AngelicaChatz';

**Q9: Find the songs contained in a playlist**

**Create** **‘playlist\_songs’** **table**  
  
CREATE TABLE playlist\_songs(  
playlist\_name text,   
song\_name text,   
PRIMARY KEY (playlist\_name,song\_name)  
)WITH COMMENT = 'Q9 Find the songs contained in a playlist';

**Insert dummy data to ‘playlist\_songs’ table**

INSERT INTO playlist\_songs (playlist\_name,song\_name)   
VALUES ('Dance Hits','Sexual Revolution');

**Query**

SELECT song\_name FROM playlist\_songs WHERE playlist\_name='Dance Hits';

**Q10: Find how many times a playlist has been played**

**Create ‘playlist\_no\_of\_times\_played’ table**CREATE TABLE playlist\_no\_of\_times\_played(  
playlist\_name text,   
times\_played int,   
PRIMARY KEY (playlist\_name)  
)WITH COMMENT = 'Q10 Find how many times a playlist has been played';

**Insert dummy data to ‘playlist\_no\_of\_times\_played’ table**INSERT INTO playlist\_no\_of\_times\_played(playlist\_name,times\_played)   
VALUES ('Dance Hits',6669);

**Query**

SELECT times\_played FROM playlist\_no\_of\_times\_played WHERE playlist\_name = 'Dance Hits';

**Q11: Find how many times a song has been played**

**Create ‘song\_no\_of\_times\_played’ table**CREATE TABLE song\_no\_of\_times\_played(  
song\_name text,  
times\_played int,  
PRIMARY KEY (song\_name)  
)WITH COMMENT = 'Q11 Find how many times a song has been played';

**Insert dummy data to ‘song\_no\_of\_times\_played’ table**INSERT INTO song\_no\_of\_times\_played (song\_name,times\_played)  
VALUES ('lost on you',1450069);

**Query**

SELECT times\_played FROM song\_no\_of\_times\_played WHERE song\_name='lost on you';

**Q12: List playlists in decreasing popularity; the popularity of a playlist is defined by the number of times it has been played.**

**Create ‘songs\_by\_user’ table**

CREATE TABLE playlist\_No\_of\_followers\_desc(  
id bigint,  
playlist\_name text,  
no\_followers int,   
PRIMARY KEY(id,no\_followers)  
) WITH COMMENT = 'Q12:List playlists in decreasing popularity';

**Insert dummy data to ‘songs\_by\_user’ table**

INSERT INTO playlist\_No\_of\_followers\_desc(id,playlist\_name,no\_followers)   
VALUES (1,'Dance Hits',32154);

INSERT INTO playlist\_No\_of\_followers\_desc(id,playlist\_name,no\_followers)  
VALUES (1,'Trip Hop',6669);

**Query**

SELECT \* FROM playlist\_No\_of\_followers\_desc WHERE id=0 ORDER BY followers DESC;

**Q13: List users in decreasing popularity: the popularity of a user is defined by the number of followers.**

**Create ‘user\_No\_of\_followers\_desc’ table**

CREATE TABLE user\_No\_of\_followers\_desc(  
id bigint,  
user\_name text,  
no\_followers int,   
PRIMARY KEY(id,no\_followers)   
) WITH COMMENT = 'Q13:List users in decreasing popularity';

**Insert dummy data to ‘user\_No\_of\_followers\_desc’ table**  
  
INSERT INTO user\_No\_of\_followers\_desc(id,user\_name,no\_followers)   
VALUES (1,'AngelicaChatz',654789);

INSERT INTO playlist\_No\_of\_followers\_desc(id,user\_name,no\_followers)   
VALUES (0,'AnastasiaRilmaheri',789654123);

**Query**SELECT \* FROM user\_No\_of\_followers\_desc WHERE id=0 ORDER BY followers DESC;