

Συστήματα Διαχείρισης Δεδομένων Μεγάλου Όγκου

Εργαστηριακή Άσκηση 2022/23

Όνομα	Επώνυμο	ΑΜ	Github	Email:
Γρηγόριος Γεώργιος	Καπαδούκας	1072484	GregKapadoukas	up1072484@upnet.gr
Κριστιάν	Λούκα	1072625	Louk4s	up1072625@upnet.gr

Βεβαιώνω ότι είμαι συγγραφέας της παρούσας εργασίας και ότι έχω αναφέρει ή παραπέμψει σε αυτήν, ρητά και συγκεκριμένα, όλες τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, προτάσεων ή λέξεων, είτε αυτές μεταφέρονται επακριβώς (στο πρωτότυπο ή μεταφρασμένες) είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για το συγκεκριμένο μάθημα/σεμινάριο/πρόγραμμα σπουδών.

Έχω ενημερωθεί ότι σύμφωνα με τον εσωτερικό κανονισμό λειτουργίας του Πανεπιστημίου Πατρών άρθρο 50§6, τυχόν προσπάθεια αντιγραφής ή εν γένει φαλκίδευσης της εξεταστικής και εκπαιδευτικής διαδικασίας από οιονδήποτε εξεταζόμενο, πέραν του μηδενισμού, συνιστά βαρύ πειθαρχικό παράπτωμα.

Υπογραφή

Γρηγόρης Καπαδούκας

21 / 09 / 2023

Υπογραφή

Κριστιάν Λούκα

21 / 09 / 2023

Συνημμένα αρχεία κώδικα

Μαζί με την παρούσα αναφορά υποβάλλουμε τα παρακάτω αρχεία κώδικα

Αρχείο	Αφορά το ερώτημα	Περιγραφή/Σχόλιο
create.cql	2	CQL script φόρτωσης της βάσης
select.cql	3	CQL εντολές SELECT για τα ερωτήματα που ζητούνται
consistency_one.ipynb	2, 3, 4	Python κώδικας (Jupyter Notebook) για προεπεξεργασία δεδομένων, φόρτωση δεδομένων (INSERT) και εκτέλεση ερωτημάτων (SELECT), με consistency level ONE. Περιέχει τα αποτελέσματα των μετρήσεων στο εργαστήριο (εκτέλεση στο microcluster)
consistency_quorum.ipynb	2, 3, 4	Το ίδιο με πάνω, μόνη αλλαγή ότι είναι για consistency level QUORUM
consistency_all.ipynb	2, 3, 4	Το ίδιο με πάνω, μόνη αλλαγή ότι είναι για consistency level ALL

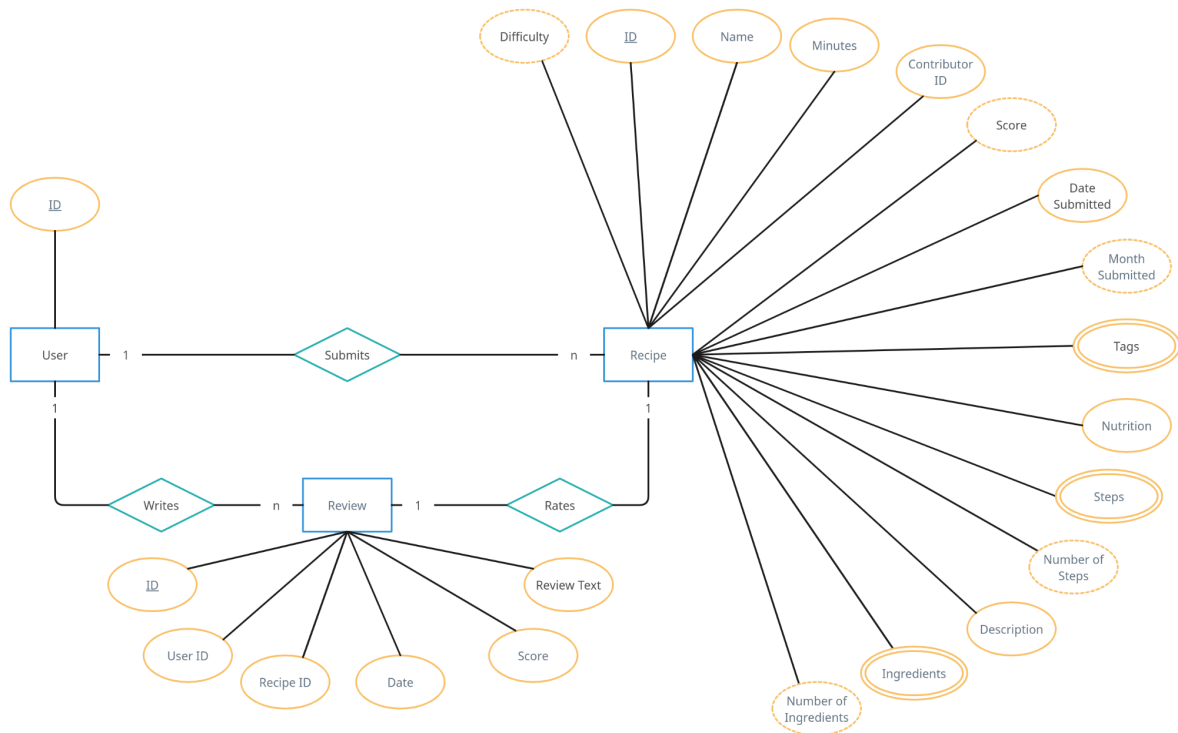
Τεχνικά χαρακτηριστικά περιβάλλοντος λειτουργίας

Χαρακτηριστικό	Τιμή
CPU model	AMD Ryzen 7 4800H
CPU clock speed	2.9 GHz
Physical CPU cores	8
Logical CPU cores	16
RAM	2*8 GiB DDR4 3.2GHz
Secondary Storage Type	NVME SSD

Για την υλοποίηση των ερωτημάτων στο προσωπικό υπολογιστή, αποφασίσαμε αντί για χρήση Ubuntu VM να στήσουμε τη Cassandra χρησιμοποιώντας Docker container. Με αυτή την επιλογή επιτυγχάνουμε κοντινότερη σε native απόδοση, καθώς και εξαλείφουμε τη περίπτωση να μας εμφανιστούν προβλήματα, λόγω μη εγκατεστημένων dependencies, ή ασύμβατων εκδόσεων προγραμμάτων.

Ερώτημα 1: Σχεδιασμός ΒΔ

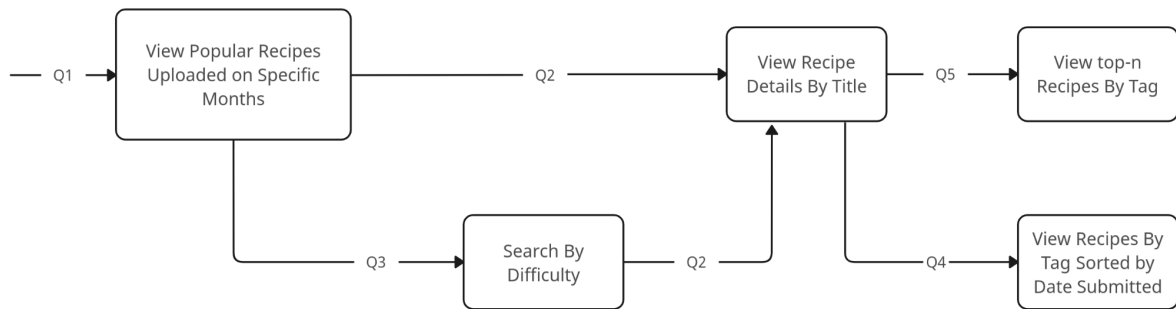
Εννοιολογικό μοντέλο:



Σχετικά με το εννοιολογικό μοντέλο σημειώνω ότι προέκυψε αναλύοντας τα δεδομένα που μας δίνονται στα αρχεία του dataset RAW_interactions.csv και RAW_recipes.csv, και ο σκοπός του είναι να αναλύσουμε τις έννοιες και τις συσχετίσεις μεταξύ των δεδομένων που μας δίνονται.

Προφανώς επίσης, λόγω της query-first διαδικασίας μοντελοποίησης που ακολουθούμε για να φτιάξουμε τη βάση μέσω Cassandra, δεν χρειαζόμαστε τελικά δεδομένα για όλες τις παραπάνω έννοιες, παρά θα καταλήξουμε να χρησιμοποιήσουμε μόνο αυτά που χρειαζόμαστε για να υποστηρίξουμε τα queries που μας ζητούνται, καθώς και σε ορισμένες περιπτώσεις που χρειάζεται να φτιάξουμε τα δικά μας δεδομένα. Αυτά τα παραγόμενα γνωρίσματα τα δείχνουμε στο εννοιολογικό μοντέλο με κύκλο με κουκίδες (παραγόμενα γνωρίσματα).

Application Workflow:



Σχετικά με το application workflow αρχικά θα εξηγήσω πως προέκυψαν τα queries (Q) με βάση τα ερωτήματα της εκφώνησης στην υποενότητα “Ερώτημα 1: Σχεδιασμός βάσης δεδομένων”:

Αρχικά το ερώτημα 1) αντιστοιχείται στο ερώτημα Q1. Έπειτα τα ερωτήματα 2) και 4) αντιστοιχούνται στο query Q2.

Αυτό συμβαίνει επειδή το ερώτημα 2) ζητά την υποστήριξη αναζήτησης συνταγών που περιέχουν κάποιες λέξεις - κλειδιά στο τίτλο. Υποθέτουμε πως εδώ ζητείται να γίνεται αναζήτηση με βάση ολόκληρο το τίτλο, επειδή η Cassandra δεν υποστηρίζει αποδοτικά queries με ερωτήματα τύπου LIKE της SQL, οπότε περιοριζόμαστε στην ισότητα του ονόματος.

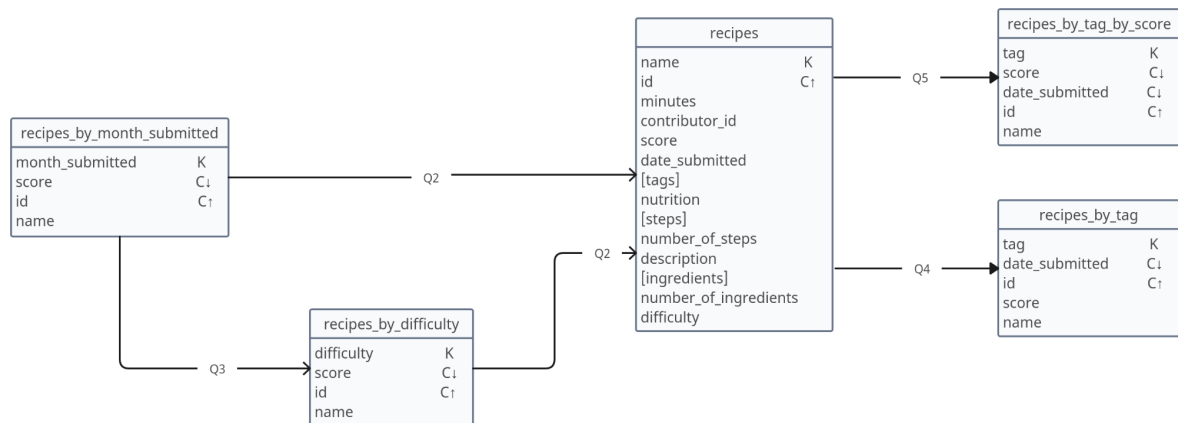
Το ερώτημα 4) από την άλλη ζητάει λεπτομερείς πληροφορίες για κάποια συνταγή. Παρατηρούμε όμως στην υποενότητα “Ερώτημα 3: Εκτέλεση ερωτημάτων” το ερώτημα “2. Εμφάνιση όλων των λεπτομερειών για την ταινία <<chick greek salad>>...”, που μας παραπέμπει ότι θα πρέπει να γίνει partitioning με βάση το όνομα, όπως χρειάζεται και για το ερώτημα 2), οπότε μας συμφέρει να θεωρήσουμε ένα query που να κάνει τη λειτουργία και του ερωτήματος 2) και 4), το οποίο είναι το Q1.

Έτσι στη τελική σχεδιάζουμε ένα πίνακα με βάση query based design που πληρεί και το ερώτημα 2) και 4).

Για το ερώτημα 3) επίσης έχουμε το query Q3. Για το ερώτημα 5) παρατηρούμε πως μας ζητείται αρχικά μέσω των ‘top-n’ συνταγών να γίνει clustering με βάση το score, με το οποίο συμφωνεί και το ‘Ερώτημα 3: Εκτέλεση ερωτημάτων: Ερώτημα 5’. Παρόλα αυτά, στο ‘Ερώτημα 3: Εκτέλεση ερωτημάτων: Ερώτημα 4’ ζητείται πάλι αναζήτηση με βάση τα tags, αλλά αυτή τη φορά συσταδοποίηση με βάση το date_submitted. Οπότε ορίζω query Q4 όταν θέλουμε αναζήτηση με βάση tags με κατάταξη του date_submitted και Q5 όταν θέλουμε αναζήτηση με βάση tags και κατάταξη του score.

Σημειώνω επίσης εδώ πως έχω κάνει τη παραδοχή τα Q4, Q5 να γίνονται πάντα μετά από ενέργεια που κάνει Q2. Αυτό το θεώρησα επειδή από την έκφραση των ερωτημάτων στην εκφώνηση θεώρησα ότι ο χρήστης πρέπει στο UI να επιλέξει μια από τις ετικέτες στη σελίδα λεπτομερειών για να του εμφανιστούν οι top-n συνταγές που σχετίζονται με την ετικέτα αυτή.

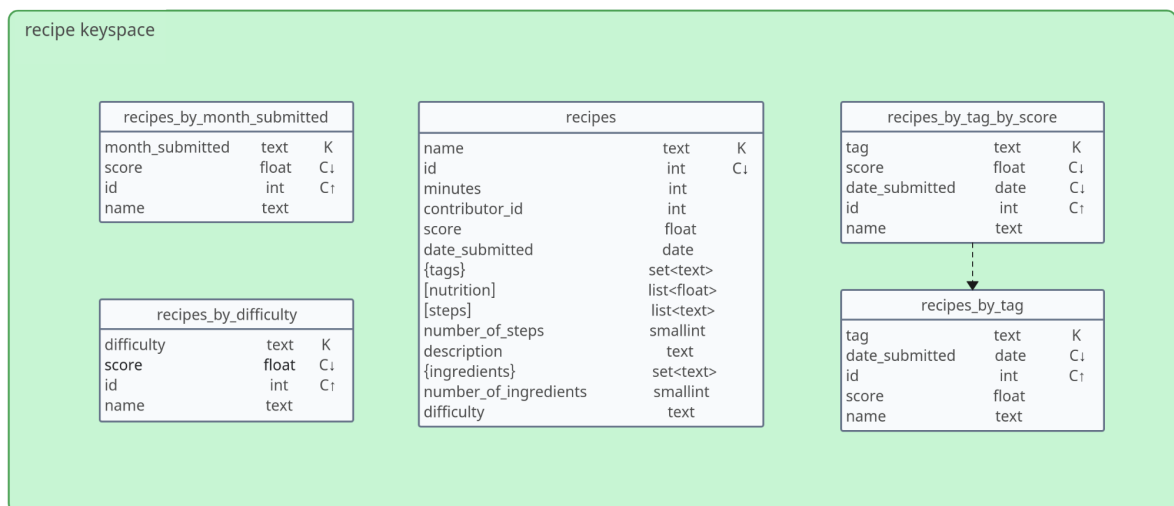
Bonus: Logical Data Model:



Ενώ το Logical Data Model δεν μας ζητείται, σύμφωνα με το βιβλίο “Cassandra The Definitive Guide - Jeff Carpenter & Eben Hewitt” (Appendix 1 στο eclass) προηγείται του σχεδιασμού του Chebotko Diagram, οπότε το συμπεριλαμβάνω και αυτό

Δεν μπαίνω όμως σε λεπτομέρεια σχεδιαστικών αποφάσεων ακόμα, εφόσον αυτό θα το κάνω παρακάτω για το πλήρες Chebotko Diagram.

Chebotko Diagram (Physical Data Model):



Σχετικά με το Chebotko diagram αναφέρω αρχικά ότι αποφάσισα να σχεδιάσω όλη τη βάση σε ένα μόνο keyspace. Αυτό συμβαίνει λόγω της δομής του application workflow, εφόσον όλα τα queries γίνονται σε μια ακολουθία πράξεων του χρήστη χωρίς να υπάρχει ενδιάμεση απόφαση για αλλαγή ενέργειας, καθώς και υπάρχει νοηματική συνέχεια μεταξύ των πινάκων.

Σημειώνω επίσης πως το score, αναφέρεται στο μέσο σκορ αξιολόγησης που προκύπτει κάνοντας aggregate τις τιμές των reviews που μας δίνεται στο dataset, μέσω κώδικα Python.

Επειδή τα queries δεν μας ζητάνε ποτέ επιμέρους αξιολογήσεις, δεν έχει νόημα να αποθηκεύσουμε κάτι άλλο εκτός του μέσου score στη Cassandra, και εφόσον αυτή δεν έχει τη δυνατότητα να κάνει aggregations στην εισαγωγή, αυτό το κάνουμε σαν preprocessing step πριν την εισαγωγή των δεδομένων.

Σχετικά με τον πίνακα `recipe_by_month_submitted`, αυτός υλοποιεί το Q1. Ο τρόπος που γίνεται αυτό είναι λόγω του partition key στο πεδίο `month_submitted`, οπότε κάθε partition περιέχει τα δεδομένα που αναρτήθηκαν σε ένα ημερολογιακό μήνα (το `month_submitted` υπολογίζεται επίσης στη Python στο preprocessing step). Επίσης το score είναι clustering column, ώστε να γίνει κατάταξη με βάση τη βαθμολογία, καθώς και το id αποτελεί clustering column ώστε να σιγουρευτώ ότι κάθε εγγραφή θα έχει μοναδικό primary key. Παραπάνω για τον πίνακα `recipe_by_month_submitted` και πως χρησιμοποιείται για την επιτέλεση του Q1 αναφέρω παρακάτω στις επεξηγήσεις για το κώδικα στο “Ερώτημα 3: Απαντήσεις ερωτημάτων” (εγγενής πρόβλημα υποστήριξης του query Q1 σε Cassandra λόγω ανάγκης συνδυασμού χρήσης IN keyword με ORDER BY - καλύτερη λύση η προσωρινή απενεργοποίηση του paging πολυ επιλύει το θέμα ή το sorting στο client side).

Σχετικά με τον πίνακα `recipe_by_difficulty`, αυτός υλοποιεί το Q3. Έχουμε αρχικά υπολογίσει τη τιμή difficulty μέσω της ενδεικτικής σχέσης που μας δίνεται ‘πλήθος βημάτων x χρόνος παρασκευής’. Η διαδικασία υπολογισμού του difficulty γίνεται πάλι σε κώδικα Python, πριν την εισαγωγή των δεδομένων στη Cassandra, εφόσον η Cassandra δεν μπορεί να κάνει πράξεις με αυτό το τρόπο. Για το θέσιμο της τιμής του difficulty ελέγξαμε τη τιμή του ‘πλήθος βημάτων x χρόνος παρασκευής’ σε ποιο από τα παρακάτω εύρη τιμών ανήκει και θέσαμε το difficulty αντίστοιχα.

- $[\min(\text{πλήθος βημάτων} \times \text{χρόνος παρασκευής}), 150] \rightarrow \text{difficulty} = \text{'easy'}$
- $[150, 350] \rightarrow \text{difficulty} = \text{'medium'}$
- $[350, 750] \rightarrow \text{difficulty} = \text{'hard'}$
- $[750, \text{difficulty.max}(\text{πλήθος βημάτων} \times \text{χρόνος παρασκευής})] \rightarrow \text{difficulty} = \text{'very hard'}$

Τέλος σημειώνω, πως θέτοντας το difficulty ως partition key καταφέρνω να έχω σε κάθε partition τις συνταγές που ανήκουν σε μια κλάση δυσκολίας, και έπειτα με το clustering column score τα κατατάσσω με βάση το score όπως μου ζητείται. Τέλος το id είναι clustering column πάλι για να εξασφαλίσω την μοναδικότητα του primary key.

Σχετικά με τον πίνακα `recipes`, αυτός υλοποιεί το Q2 (ερωτήματα 2), 4)). Έχω ορίσει το name ως partition key ώστε να δημιουργούνται partitions με βάση το όνομα, και έχω ορίσει το id ως clustering column πάλι για να εξασφαλίσω την μοναδικότητα του primary key, εφόσον υπάρχουν συνταγές με το ίδιο όνομα. Η αλλαγή μεταξύ των ερωτημάτων 2) και 4), είναι ότι για το 2) δεν χρειάζεται να κάνω select όλες τις στήλες, και στο 4) πρέπει να επιλέξω όλες τις λεπτομέρειες καθώς και να κάνω ‘LIMIT 1’ ώστε να εμφανιστεί μόνο μια συνταγή στη περίπτωση πολλών με κοινό όνομα, εφόσον εμφανίζω πληροφορίες συνταγής και δεν κάνω αναζήτηση.

Σχετικά με τον πίνακα `recipes_by_tag`, αυτός υλοποιεί το Q4. Έχω πλέον ορίσει το tag να είναι το partition key. Άρα για κάθε tag που εμφανίζεται στο dataset, υπάρχει ένα partition που περιέχει όλες τις συνταγές που έχουν αυτό το tag. Προφανώς, επειδή κάθε συνταγή έχει πολλά tags, θα υπάρχουν πολλές διπλότυπες συνταγές, αυτό όμως δεν προκαλεί πρόβλημα στη μοντελοποίηση για Cassandra. Το date submitted αποτελεί clustering column εφόσον

ζητείται στο 'Ερώτημα 3: Εκτέλεση ερωτημάτων Ερώτημα 4' να γίνει κατάταξη ανά ημερομηνία προσθήκης. Επίσης το id είναι clustering column για να εξασφαλίσουμε μοναδικότητα στο primary key.

Τέλος υπάρχει και ένα materialized view στο recipes_by_tag, το recipes_by_tag_by_score, το οποίο προσθέτει ένα clustering column στο score, επειδή στο 'Ερώτημα 3: Εκτέλεση ερωτημάτων Ερώτημα 5' μας ζητείται κατάταξη με βάση τη μέση βαθμολογία' (το score που αποθηκεύουμε είναι η μέση εξαρχής). Σημειώνω πως η προσθήκη επιπλέον πίνακα για το 'Ερώτημα 3: Εκτέλεση ερωτημάτων Ερώτημα 5' δεν έχει νόημα, εφόσον ο πίνακας recipes_by_tag περιέχει ήδη τις στήλες που χρειάζονται με τη μοντελοποίηση που έχουμε κάνει, οπότε με τη χρήση materialized views αποφεύγουμε τη ανάγκη για δέσμευση περισσότερου χώρου στα nodes του cluster, καθώς και πολύ μεγαλύτερα insertion times (το recipes_by_tag είναι μακράν ο μεγαλύτερος πίνακας).

Σχετικά με την επιλογή μου να προσθέσω το materialized view στο πεδίο score, σημειώνω ότι θα μπορούσα αντίστοιχα να είχα clustering column το score στο recipes_by_tag και να φτιάξω materialized view recipes_by_tag_by_date_submitted, υποστηρίζοντας τα ίδια queries στη τελική. Σύμφωνα με το βιβλίο "Cassandra The Definitive Guide - Jeff Carpenter & Eben Hewitt" (Appendix 1 στο eclass), θέλουμε στα columns τα οποία επιλέγουμε να προσθέσουμε clustering μέσω materialized view να έχουν υψηλό cardinality τιμών, οπότε υποθέτω πως επειδή το score είναι πραγματικός αριθμός ότι θα έχει υψηλότερο cardinality, αν και αυτό δεν το γνωρίζω σίγουρα, στη συγκεκριμένη περίπτωση η διαφορά απόδοσης μεταξύ των προσεγγίσεων θα είναι μικρή έτσι και αλλιώς.

Τέλος σημειώνω πως μερικοί πίνακες έχουν το πεδίο score και ας μην το χρειάζονται για τα ερωτήματα, αυτό γίνεται επειδή καθαρά από μεριάς UI design επειδή υποθέτουμε πως το score είναι μια πληροφορία που θα θέλει ο χρήστης να βλέπει κάνοντας τα συγκεκριμένα queries μέσω της εφαρμογής.

Ερώτημα 2: Ερωτήματα DDL

Keyspace	recipe
DDL statement	CREATE KEYSPACE recipe WITH replication = {'class': 'SimpleStrategy', 'replication_factor' : 3};

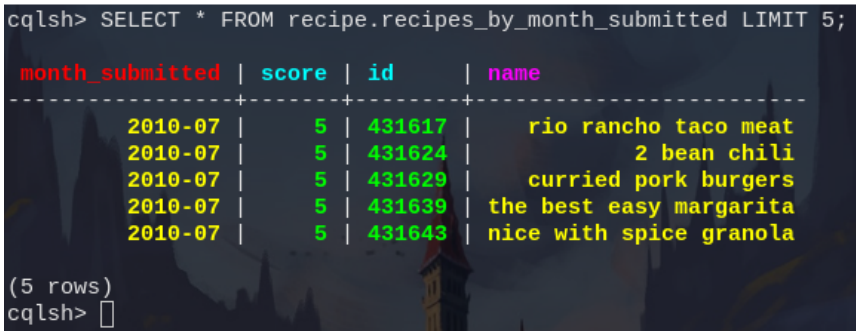
Table	recipe.recipes_by_month_submitted
DDL statement	CREATE TABLE recipe.recipes_by_month_submitted (month_submitted text, score float, id int, name text, PRIMARY KEY ((month_submitted), score, id)) WITH CLUSTERING ORDER BY (score DESC, id ASC);
Screenshot	

Table	recipe.recipes_by_difficulty
DDL statement	CREATE TABLE recipe.recipes_by_difficulty (

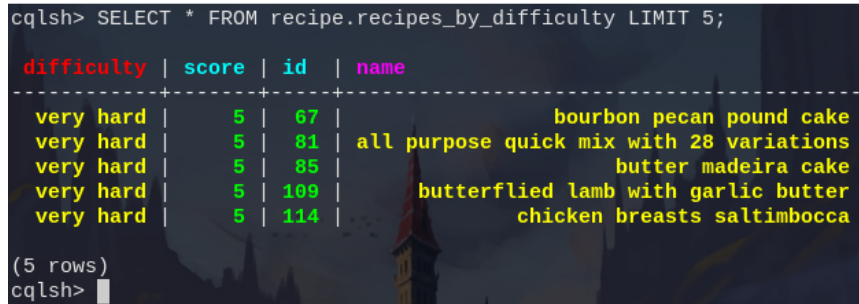
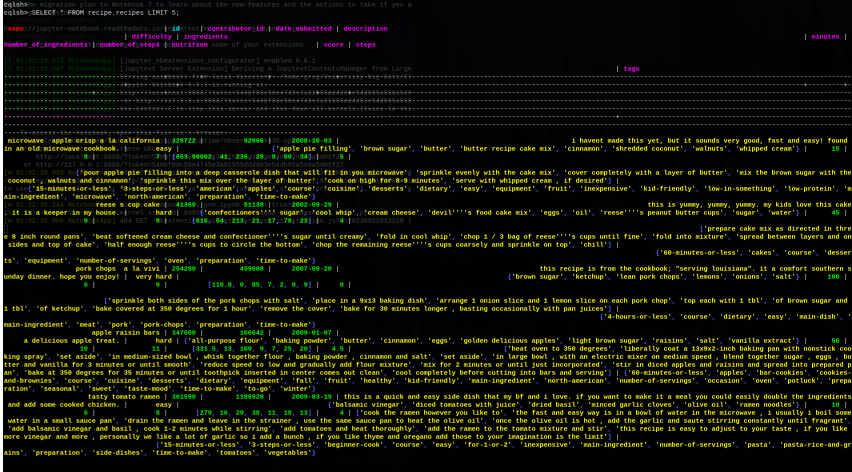
	<p>difficulty text,</p> <p>score float,</p> <p>id int,</p> <p>name text,</p> <p>PRIMARY KEY (difficulty, score, id)</p> <p>) WITH CLUSTERING ORDER BY (score DESC, id ASC);</p>																								
Screenshot	 <pre>cqlsh> SELECT * FROM recipe.recipes_by_difficulty LIMIT 5;</pre> <table><thead><tr><th>difficulty</th><th>score</th><th>id</th><th>name</th></tr></thead><tbody><tr><td>very hard</td><td>5</td><td>67</td><td>bourbon pecan pound cake</td></tr><tr><td>very hard</td><td>5</td><td>81</td><td>all purpose quick mix with 28 variations</td></tr><tr><td>very hard</td><td>5</td><td>85</td><td>butter madeira cake</td></tr><tr><td>very hard</td><td>5</td><td>109</td><td>butterflied lamb with garlic butter</td></tr><tr><td>very hard</td><td>5</td><td>114</td><td>chicken breasts saltimbocca</td></tr></tbody></table> <p>(5 rows) cqlsh></p>	difficulty	score	id	name	very hard	5	67	bourbon pecan pound cake	very hard	5	81	all purpose quick mix with 28 variations	very hard	5	85	butter madeira cake	very hard	5	109	butterflied lamb with garlic butter	very hard	5	114	chicken breasts saltimbocca
difficulty	score	id	name																						
very hard	5	67	bourbon pecan pound cake																						
very hard	5	81	all purpose quick mix with 28 variations																						
very hard	5	85	butter madeira cake																						
very hard	5	109	butterflied lamb with garlic butter																						
very hard	5	114	chicken breasts saltimbocca																						

Table	recipe.recipes
DDL statement	<pre>CREATE TABLE recipe.recipes(name text, id int, minutes int, contributor_id int, score float, date_submitted date, tags set<text>, nutrition list<float>, steps list<text>, number_of_steps smallint, description text,</pre>

	<div>ingredients set<text>, number_of_ingredients smallint, difficulty text, PRIMARY KEY ((name), id)) WITH CLUSTERING ORDER BY (id ASC);</div>
<div>Screenshot (παρακαλώ κάντε zoom, η ευκρίνεια είναι υψηλή και τα γράμματα φαίνονται έτσι. Δυστυχώς δεν μπορώ να τα χωρέσω με πιο όμορφο τρόπο.)</div>	<div></div>


<div>Table</div>	<div>recipe.recipes_by_tag</div>
<div>DDL statement</div>	<div>CREATE TABLE recipe.recipes_by_tag (tag text, date_submitted date, id int, score float, name text, PRIMARY KEY ((tag), date_submitted, id)) WITH CLUSTERING ORDER BY (date_submitted DESC, id ASC);</div>

Screenshot

```
cqlsh> SELECT * FROM recipe.recipes_by_tag LIMIT 5;
```

tag	date_submitted	id	name	score
malaysian	2013-03-22	497993	laksa flavoured prawns shrimp and hokkien noodles	5
malaysian	2012-01-17	472074	burmese breakfast fried rice	0
malaysian	2011-12-19	470178	malaysian lime coconut swordfish	4
malaysian	2011-09-16	464232	mee goreng tauceo stir fried noodles with shrimp and greens	5
malaysian	2011-08-29	463234	nasi minyak oil rice using basmathi rice	5

```
(5 rows)
cqlsh>
```

Table	recipe.recipes_by_tag_by_score																														
DDL statement (Materialized View)	CREATE MATERIALIZED VIEW recipe.recipes_by_tag_by_score AS SELECT * FROM recipe.recipes_by_tag WHERE tag IS NOT NULL AND date_submitted IS NOT NULL AND score IS NOT NULL AND id IS NOT NULL AND name IS NOT NULL PRIMARY KEY ((tag), score, date_submitted, id);																														
Screenshot	 <pre>cqlsh> SELECT * FROM recipe.recipes_by_tag_by_score LIMIT 5;</pre> <table><thead><tr><th>tag</th><th>score</th><th>date_submitted</th><th>id</th><th>name</th></tr></thead><tbody><tr><td>malaysian</td><td>5</td><td>2013-03-22</td><td>497993</td><td>laksa flavoured prawns shrimp and hokkien noodles</td></tr><tr><td>malaysian</td><td>5</td><td>2011-09-16</td><td>464232</td><td>mee goreng tauceo stir fried noodles with shrimp and greens</td></tr><tr><td>malaysian</td><td>5</td><td>2011-08-29</td><td>463234</td><td>nasi minyak oil rice using basmathi rice</td></tr><tr><td>malaysian</td><td>5</td><td>2011-05-06</td><td>455269</td><td>astragalus vegetable soup detox soup</td></tr><tr><td>malaysian</td><td>5</td><td>2010-05-12</td><td>424763</td><td>malaysian baked papaya with ginger</td></tr></tbody></table> <p>(5 rows) cqlsh> </p>	tag	score	date_submitted	id	name	malaysian	5	2013-03-22	497993	laksa flavoured prawns shrimp and hokkien noodles	malaysian	5	2011-09-16	464232	mee goreng tauceo stir fried noodles with shrimp and greens	malaysian	5	2011-08-29	463234	nasi minyak oil rice using basmathi rice	malaysian	5	2011-05-06	455269	astragalus vegetable soup detox soup	malaysian	5	2010-05-12	424763	malaysian baked papaya with ginger
tag	score	date_submitted	id	name																											
malaysian	5	2013-03-22	497993	laksa flavoured prawns shrimp and hokkien noodles																											
malaysian	5	2011-09-16	464232	mee goreng tauceo stir fried noodles with shrimp and greens																											
malaysian	5	2011-08-29	463234	nasi minyak oil rice using basmathi rice																											
malaysian	5	2011-05-06	455269	astragalus vegetable soup detox soup																											
malaysian	5	2010-05-12	424763	malaysian baked papaya with ginger																											

Ερώτημα 3: Απαντήσεις ερωτημάτων

<div>Ερώτημα</div> <div>Εμφάνιση των 30 συνταγών με την υψηλότερη μέση βαθμολογία μεταξύ 01/01/2012 και 31/05/2012</div> <div>(Ζητείται να εμφανιστούν μόνο οι πρώτες 5)</div> <div>(Παρακάτω επεξηγώ την αναγκαιότητα προσωρινής απενεργοποίησης του PAGING)</div>	<div>Απάντηση</div> <div><pre>cqlsh> PAGING OFF; Disabled Query paging. cqlsh> SELECT name, month_submitted, score ... FROM recipe.recipes_by_month_submitted ... WHERE month_submitted IN ('2012-01', '2012-02', '2012-03', '2012-04', '2012-05') ... ORDER BY score DESC ... LIMIT 30; PAGING ON;</pre><table><thead><tr><th>name</th><th>month_submitted</th><th>score</th></tr></thead><tbody><tr><td>potato chip sandwich</td><td>2012-01</td><td>5</td></tr><tr><td>chip butty</td><td>2012-01</td><td>5</td></tr><tr><td>chic greek oven pork chops with hominy</td><td>2012-01</td><td>5</td></tr><tr><td>italian pound cake food processor</td><td>2012-01</td><td>5</td></tr><tr><td>tomato mushroom soup</td><td>2012-01</td><td>5</td></tr><tr><td>apple pomegranate and wheat berry salad</td><td>2012-01</td><td>5</td></tr><tr><td>florentine breakfast wraps</td><td>2012-01</td><td>5</td></tr><tr><td>baum torte baum kuchen german tree cake</td><td>2012-01</td><td>5</td></tr><tr><td>homemade granola</td><td>2012-01</td><td>5</td></tr><tr><td>maw maw gilman's peanut banana salad</td><td>2012-01</td><td>5</td></tr><tr><td>lebanese chicken and potatoes</td><td>2012-01</td><td>5</td></tr><tr><td>middle eastern arugula salad</td><td>2012-01</td><td>5</td></tr><tr><td>vegetable medley with toasted sesame seeds</td><td>2012-01</td><td>5</td></tr><tr><td>thai baked salmon</td><td>2012-01</td><td>5</td></tr><tr><td>really really good and easy buttercream frosting</td><td>2012-01</td><td>5</td></tr><tr><td>red curry garlic hummus</td><td>2012-01</td><td>5</td></tr><tr><td>roasted sirloin beef weight watchers</td><td>2012-01</td><td>5</td></tr><tr><td>herbed rib roast weight watchers</td><td>2012-01</td><td>5</td></tr><tr><td>izakaya sakura teriyaki chicken</td><td>2012-01</td><td>5</td></tr><tr><td>christmas bagel bites with leek spread and smoked salmon</td><td>2012-01</td><td>5</td></tr><tr><td>decadent vegan chocolate mousse</td><td>2012-01</td><td>5</td></tr><tr><td>middle eastern feta spread</td><td>2012-01</td><td>5</td></tr><tr><td>slow cooker spinach lasagna</td><td>2012-01</td><td>5</td></tr><tr><td>cheese capers</td><td>2012-01</td><td>5</td></tr><tr><td>mandalay bay fruit salad</td><td>2012-01</td><td>5</td></tr><tr><td>grilled sea scallops</td><td>2012-01</td><td>5</td></tr><tr><td>beets with lemon sauce</td><td>2012-01</td><td>5</td></tr><tr><td>indonesian style chicken curry</td><td>2012-01</td><td>5</td></tr><tr><td>asian tuna and cabbage salad</td><td>2012-01</td><td>5</td></tr><tr><td>wilted greens with bacon vinaigrette</td><td>2012-01</td><td>5</td></tr></tbody></table><div>(30 rows) cqlsh> PAGING ON; Now Query paging is enabled Page size: 100 cqlsh></div></div>	name	month_submitted	score	potato chip sandwich	2012-01	5	chip butty	2012-01	5	chic greek oven pork chops with hominy	2012-01	5	italian pound cake food processor	2012-01	5	tomato mushroom soup	2012-01	5	apple pomegranate and wheat berry salad	2012-01	5	florentine breakfast wraps	2012-01	5	baum torte baum kuchen german tree cake	2012-01	5	homemade granola	2012-01	5	maw maw gilman's peanut banana salad	2012-01	5	lebanese chicken and potatoes	2012-01	5	middle eastern arugula salad	2012-01	5	vegetable medley with toasted sesame seeds	2012-01	5	thai baked salmon	2012-01	5	really really good and easy buttercream frosting	2012-01	5	red curry garlic hummus	2012-01	5	roasted sirloin beef weight watchers	2012-01	5	herbed rib roast weight watchers	2012-01	5	izakaya sakura teriyaki chicken	2012-01	5	christmas bagel bites with leek spread and smoked salmon	2012-01	5	decadent vegan chocolate mousse	2012-01	5	middle eastern feta spread	2012-01	5	slow cooker spinach lasagna	2012-01	5	cheese capers	2012-01	5	mandalay bay fruit salad	2012-01	5	grilled sea scallops	2012-01	5	beets with lemon sauce	2012-01	5	indonesian style chicken curry	2012-01	5	asian tuna and cabbage salad	2012-01	5	wilted greens with bacon vinaigrette	2012-01	5
name	month_submitted	score																																																																																												
potato chip sandwich	2012-01	5																																																																																												
chip butty	2012-01	5																																																																																												
chic greek oven pork chops with hominy	2012-01	5																																																																																												
italian pound cake food processor	2012-01	5																																																																																												
tomato mushroom soup	2012-01	5																																																																																												
apple pomegranate and wheat berry salad	2012-01	5																																																																																												
florentine breakfast wraps	2012-01	5																																																																																												
baum torte baum kuchen german tree cake	2012-01	5																																																																																												
homemade granola	2012-01	5																																																																																												
maw maw gilman's peanut banana salad	2012-01	5																																																																																												
lebanese chicken and potatoes	2012-01	5																																																																																												
middle eastern arugula salad	2012-01	5																																																																																												
vegetable medley with toasted sesame seeds	2012-01	5																																																																																												
thai baked salmon	2012-01	5																																																																																												
really really good and easy buttercream frosting	2012-01	5																																																																																												
red curry garlic hummus	2012-01	5																																																																																												
roasted sirloin beef weight watchers	2012-01	5																																																																																												
herbed rib roast weight watchers	2012-01	5																																																																																												
izakaya sakura teriyaki chicken	2012-01	5																																																																																												
christmas bagel bites with leek spread and smoked salmon	2012-01	5																																																																																												
decadent vegan chocolate mousse	2012-01	5																																																																																												
middle eastern feta spread	2012-01	5																																																																																												
slow cooker spinach lasagna	2012-01	5																																																																																												
cheese capers	2012-01	5																																																																																												
mandalay bay fruit salad	2012-01	5																																																																																												
grilled sea scallops	2012-01	5																																																																																												
beets with lemon sauce	2012-01	5																																																																																												
indonesian style chicken curry	2012-01	5																																																																																												
asian tuna and cabbage salad	2012-01	5																																																																																												
wilted greens with bacon vinaigrette	2012-01	5																																																																																												
<div>Εμφάνιση όλων των λεπτομερειών για την ταινία «chic greek salad» (κατηγορία δυσκολίας, διατροφικές αξίες, βήματα, περιγραφή, μέση βαθμολογία)</div> <div>(Το «chick greek salad» δεν υπάρχει στο dataset, δείχνω το «greek salad»)</div>	<div><pre>cqlsh> SELECT name, difficulty, nutrition, steps, description, score FROM recipe.recipes WHERE name = 'greek salad' LIMIT 1;</pre><table><thead><tr><th>name</th><th>difficulty</th><th>nutrition</th><th>steps</th><th>description</th><th>score</th></tr></thead><tbody><tr><td>greek salad</td><td>easy</td><td>150 g, 25 g, 15 g, 10 g, 20 g, 10 g</td><td>1. Dice green pepper, red pepper, and yellow pepper. onion, cucumber and tomatoes. 2. halve the olives and remove pits if not already done. 3. toss in a large bowl with the vinaigrette dressing, feta cheese and a splash of balsamic vinegar. 4. I used to make this when I was on weight-watchers program. It is very tasty for lunch and is quite filling with an english muffin, some yogurt, and fruit.</td><td></td><td>5</td></tr></tbody></table><div>(1 rows) cqlsh></div></div>	name	difficulty	nutrition	steps	description	score	greek salad	easy	150 g, 25 g, 15 g, 10 g, 20 g, 10 g	1. Dice green pepper, red pepper, and yellow pepper. onion, cucumber and tomatoes. 2. halve the olives and remove pits if not already done. 3. toss in a large bowl with the vinaigrette dressing, feta cheese and a splash of balsamic vinegar. 4. I used to make this when I was on weight-watchers program. It is very tasty for lunch and is quite filling with an english muffin, some yogurt, and fruit.		5																																																																																	
name	difficulty	nutrition	steps	description	score																																																																																									
greek salad	easy	150 g, 25 g, 15 g, 10 g, 20 g, 10 g	1. Dice green pepper, red pepper, and yellow pepper. onion, cucumber and tomatoes. 2. halve the olives and remove pits if not already done. 3. toss in a large bowl with the vinaigrette dressing, feta cheese and a splash of balsamic vinegar. 4. I used to make this when I was on weight-watchers program. It is very tasty for lunch and is quite filling with an english muffin, some yogurt, and fruit.		5																																																																																									
<div>Εμφάνιση των ταινιών της κατηγορίας «εύκολη» ταξινομημένες ως προς τη μέση βαθμολογία τους</div> <div>(Ζητείται να εμφανιστούν μόνο οι πρώτες 5)</div>	<div><pre>cqlsh> SELECT name, difficulty, score FROM recipe.recipes_by_difficulty ... WHERE difficulty='easy' ... LIMIT 5;</pre><table><thead><tr><th>name</th><th>difficulty</th><th>score</th></tr></thead><tbody><tr><td>a jad cucumber pickle</td><td>easy</td><td>5</td></tr><tr><td>cafe cappuccino</td><td>easy</td><td>5</td></tr><tr><td>caramel apple milkshakes</td><td>easy</td><td>5</td></tr><tr><td>cherry sandwich maker snack</td><td>easy</td><td>5</td></tr><tr><td>cinnamon curry rice</td><td>easy</td><td>5</td></tr></tbody></table><div>(5 rows) cqlsh></div></div>	name	difficulty	score	a jad cucumber pickle	easy	5	cafe cappuccino	easy	5	caramel apple milkshakes	easy	5	cherry sandwich maker snack	easy	5	cinnamon curry rice	easy	5																																																																											
name	difficulty	score																																																																																												
a jad cucumber pickle	easy	5																																																																																												
cafe cappuccino	easy	5																																																																																												
caramel apple milkshakes	easy	5																																																																																												
cherry sandwich maker snack	easy	5																																																																																												
cinnamon curry rice	easy	5																																																																																												
<div>Εμφάνιση των συνταγών που περιέχουν την ετικέτα “slow-cooker” με ταξινόμηση ανά ημερομηνία προσθήκης (πιο πρόσφατες πρώτα)</div> <div>(Ζητείται να εμφανιστούν μόνο οι πρώτες 5)</div>	<div><pre>cqlsh> SELECT name, tag, date_submitted ... FROM recipe.recipes_by_tag ... WHERE tag = 'crock-pot-slow-cooker' ... ORDER BY date_submitted ... LIMIT 5;</pre><table><thead><tr><th>name</th><th>tag</th><th>date_submitted</th></tr></thead><tbody><tr><td>crock pot curried carrots</td><td>crock-pot-slow-cooker</td><td>1999-08-07</td></tr><tr><td>crock pot chocolate zucchini nut bread</td><td>crock-pot-slow-cooker</td><td>1999-08-10</td></tr><tr><td>all purpose crock pot chicken</td><td>crock-pot-slow-cooker</td><td>1999-08-10</td></tr><tr><td>barbecued pork strips</td><td>crock-pot-slow-cooker</td><td>1999-08-11</td></tr><tr><td>slow cooker bbq pinto beans</td><td>crock-pot-slow-cooker</td><td>1999-08-14</td></tr></tbody></table><div>(5 rows) cqlsh></div></div>	name	tag	date_submitted	crock pot curried carrots	crock-pot-slow-cooker	1999-08-07	crock pot chocolate zucchini nut bread	crock-pot-slow-cooker	1999-08-10	all purpose crock pot chicken	crock-pot-slow-cooker	1999-08-10	barbecued pork strips	crock-pot-slow-cooker	1999-08-11	slow cooker bbq pinto beans	crock-pot-slow-cooker	1999-08-14																																																																											
name	tag	date_submitted																																																																																												
crock pot curried carrots	crock-pot-slow-cooker	1999-08-07																																																																																												
crock pot chocolate zucchini nut bread	crock-pot-slow-cooker	1999-08-10																																																																																												
all purpose crock pot chicken	crock-pot-slow-cooker	1999-08-10																																																																																												
barbecued pork strips	crock-pot-slow-cooker	1999-08-11																																																																																												
slow cooker bbq pinto beans	crock-pot-slow-cooker	1999-08-14																																																																																												

(Το «slow-cooker» δεν υπάρχει στο dataset, δείχνω το «crock-pot-slow-cooker»)																			
Εμφάνιση των 20 συνταγών με την υψηλότερη μέση βαθμολογία για την ετικέτα “cocktail”.	<pre>cqlsh> SELECT name, tag, score ... FROM recipe.recipes_by_tag_by_score ... WHERE tag = 'cocktails' ... LIMIT 5;</pre>																		
(Ζητείται να εμφανιστούν μόνο οι πρώτες 5)	<table><thead><tr><th>name</th><th>tag</th><th>score</th></tr></thead><tbody><tr><td>bailey s chocotini</td><td>cocktails</td><td>5</td></tr><tr><td>island holiday</td><td>cocktails</td><td>5</td></tr><tr><td>masala martini</td><td>cocktails</td><td>5</td></tr><tr><td>cariad rhudd crimson love</td><td>cocktails</td><td>5</td></tr><tr><td>kaiserspritzer</td><td>cocktails</td><td>5</td></tr></tbody></table>	name	tag	score	bailey s chocotini	cocktails	5	island holiday	cocktails	5	masala martini	cocktails	5	cariad rhudd crimson love	cocktails	5	kaiserspritzer	cocktails	5
name	tag	score																	
bailey s chocotini	cocktails	5																	
island holiday	cocktails	5																	
masala martini	cocktails	5																	
cariad rhudd crimson love	cocktails	5																	
kaiserspritzer	cocktails	5																	
(Το «cocktail» δεν υπάρχει στο dataset, δείχνω το «cocktails»)	<pre>(5 rows) cqlsh> </pre>																		

Σχετικά με το πρώτο ερώτημα, δίνω έμφαση στη χρήση των ‘PAGING OFF;’ ‘PAGING ON;’ που εκτελώ στην αρχή και στο τέλος του query. Αυτό είναι αναγκαίο επειδή στη περίπτωση αυτή χρησιμοποιώ το keyword IN για να φορτώσω δεδομένα πολλών μηνών. Ενώ τα partitions που φορτώνονται είναι clustered με βάση το score εσωτερικά, η Cassandra σε αυτή τη περίπτωση θα αναγκαστεί να κάνει cluster ξανά τα αποτελέσματα των 5 partitions που φόρτωσε. Για αυτό είναι αναγκαία η χρήση του ‘ORDER BY score DESC’ στην εντολή.

Όμως, η Cassandra από default χρησιμοποιεί paging για να φορτώσει incrementally τα δεδομένα από τα partitions που ζητούνται κάθε φορά. Αυτό έχει ως συνέπεια, για να μπορέσει να γίνει το τελικό ORDER BY να πρέπει να φορτωθούν όλα τα δεδομένα εξαρχής. Αυτό επιτυγχάνεται με τη προσωρινή απενεργοποίηση του paging (αλλιώς δημιουργείται error). Δυστυχώς αυτό είναι κάτι που δεν γίνεται να αποφευχθεί λόγω της εγγενούς λειτουργίας της Cassandra, και η μόνη άλλη εφικτή προσέγγιση είναι η εξάλειψη του ORDER BY με σκοπό το sorting στο user client των αποτελεσμάτων. Διαφορετικές προσεγγίσεις μοντελοποίησης της βάσης δεν μπορούν να προσπεράσουν αυτό το περιορισμό, τουλάχιστον κατά τη δική μου γνώση.

Σημειώνω επίσης πως στη Python, εκτελώντας CQL εντολές, δεν υπάρχει η δυνατότητα να εκτελεστεί εντολή που ενεργοποιεί ή απενεργοποιεί το paging (βγάζει αυτόματα error). Κανονικά ο τρόπος χειρισμού του paging στη Python στο DataStax driver είναι η χρήση της παραμέτρου fetch_size, παρόλα αυτά αλλάζοντας τη τιμή του ο Cassandra server δεν αντιλαμβάνεται “κλειστό” το paging και δίνει error. Οπότε αναγκαστικά για τις μετρήσεις μέσω Python για το ερώτημα 1, κάνω μόνο SELECT για τις εγγραφές ενός μήνα ανά φορά.

Ερώτημα 4Α: Χρόνοι εισαγωγής δεδομένων

Δυστυχώς κατά τη διάρκεια των μετρήσεων (μέσω της εκτέλεσης των Python scripts που ετοιμάσαμε) στο εργαστηριακό χώρο μας παρουσιάστηκε το παρακάτω error:

```
WriteTimeout: Error from server: code=1100 [Coordinator node timed out waiting for replica nodes' responses] message = "Operation timed out - received only 0 responses."  
info={'consistency': 'ONE', 'required_responses': 1, 'received_responses': 0, 'write_type': 'SIMPLE'}
```

Αυτό γινόταν για κάθε consistency level (προφανώς με διαφορετικές τιμές 'received_responses' και 'required_responses' κάθε φορά).

Το συγκεκριμένο error υποθέτω πως εμφανίζεται λόγω περιορισμένων υπολογιστικών πόρων των Raspberry Pi στο microcluster, τα οποία δεν μπορούσαν να ανταπεξέλθουν στο μεγάλο υπολογιστικό φόρτο.

Σημειώνω επίσης πως σε δοκιμές που έχω κάνει χρησιμοποιώντας άλλο cluster (cluster από docker containers), οι εντολές μου όλες εκτελούνται κανονικά χωρίς προβλήματα λόγω timeout.

Επίσης δοκίμασα στο εργαστήριο να αυξήσω σημαντικά την τιμή του 'write_request_timeout_in_ms' στο cassandra.yaml configuration file όλων των Raspberry Pi (από 2000ms έφτασα τα 40000ms), σε συνδυασμό με το να μειώσω τη παράμετρο concurrent στην Python εντολή execute_concurrent (για concurrent insertions δεδομένων) από τη default τιμή 100 μέχρι και 5, αλλά ενώ οι αλλαγές έκαναν το πρόβλημα πιο σπάνιο δεν το επίλυσαν.

Για αυτό το λόγο μας προτάθηκε στο εργαστήριο, να κάνουμε τις μετρήσεις χρησιμοποιώντας ένα υποσύνολο των δεδομένων του dataset (χρησιμοποίησα τις πρώτες 30000 εγγραφές, λίγο πάνω από το 10% του dataset), ώστε να υπάρξει μικρότερη πιθανότητα να εμφανιστεί το error, και αν εμφανιστεί η πρόοδος που θα χαθεί να είναι σχετικά μικρή και να μπορέσω μέσω συνεχών δοκιμών να εκτελέσω τα scripts μέχρι να τελειώσουν επιτυχώς.

Οπότε οι παρακάτω μετρήσεις αντιστοιχούν στην εκτέλεση με 30000 εγγραφές στο microcluster των 6 Raspberry Pi.

Format: hh:mm:ss.(microseconds)	Επίπεδο write consistency (replication factor = 3)		
	ALL	QUORUM	ONE

recipes_by_month_submitted	00:00:15.920773	00:00:15.494791	00:00:18.486606
recipes_by_difficulty	00:00:22.547099	00:00:17.894894	00:00:20.020682
recipes	00:00:30.639319	00:00:43.073916	00:00:23.209049
recipes_by_tags	00:05:47.807307	00:05:12.765662	00:06:49.352167
Μέσος όρος	00:01:44.228624	00:01:37.307316	00:01:57.767126

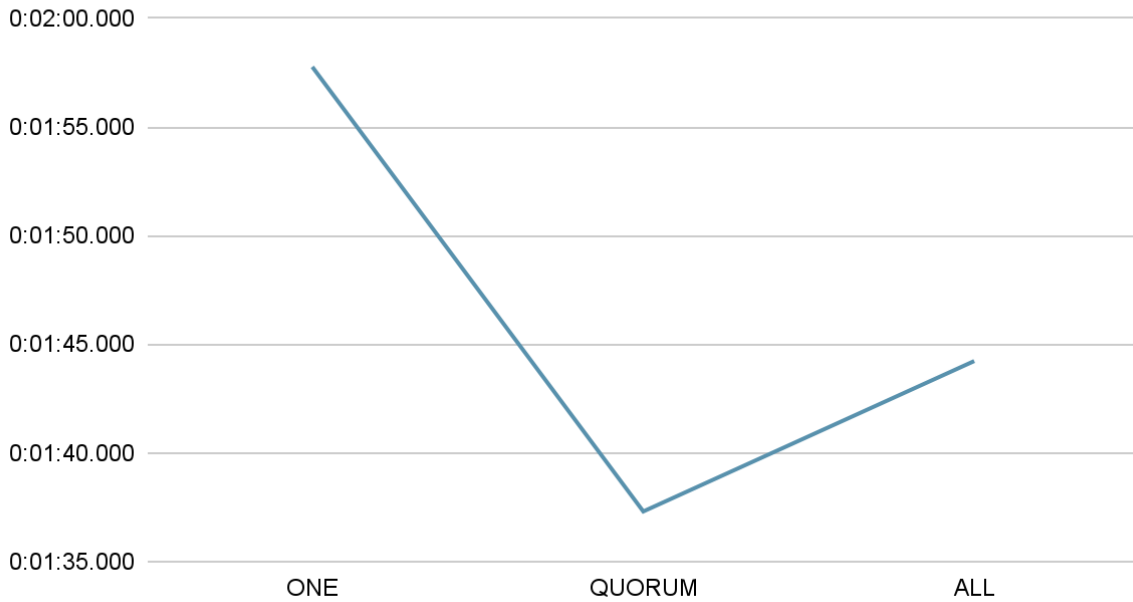
Για το materialized view recipes_by_tags_by_score δεν κάνουμε insert, δημιουργείται αυτόματα από το recipes_by_tags.

Ερώτημα 4B: Χρόνοι ανάκτησης δεδομένων

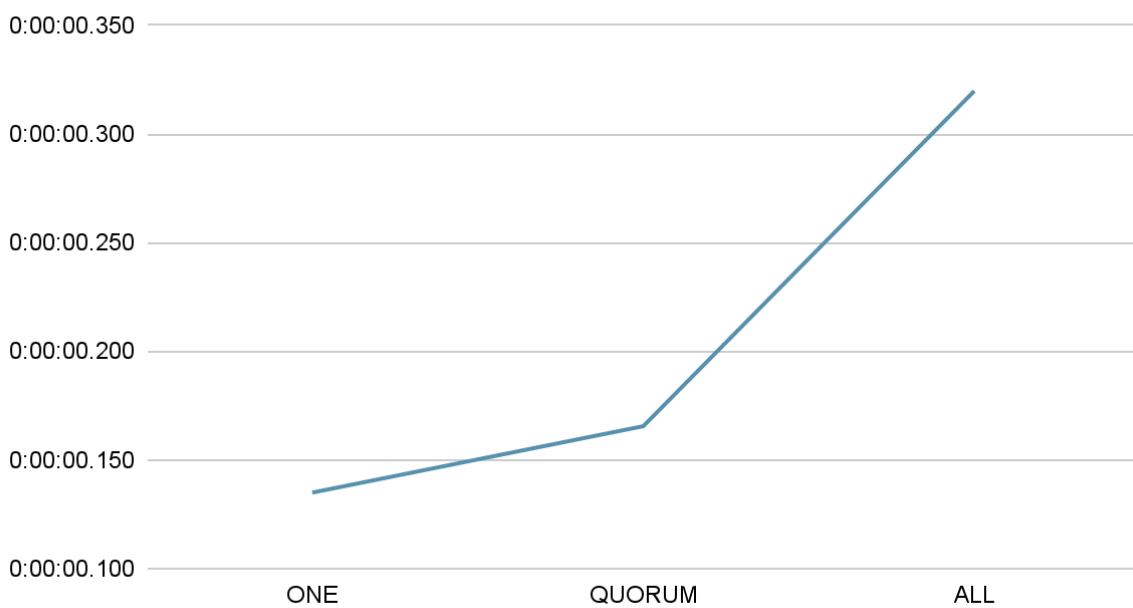
Format: hh:mm:ss.(microseconds)	Επίπεδο read consistency (replication factor = 3)		
	ALL	QUORUM	ONE
Ερώτημα 1	00:00:00.016135	00:00:00.023262	00:00:00.024661
Ερώτημα 2	00:00:00.021463	00:00:00.011079	00:00:00.045228
Ερώτημα 3	00:00:00.379271	00:00:00.233474	00:00:00.292922
Ερώτημα 4	00:00:01.153138	00:00:00.408819	00:00:00.293332
Ερώτημα 5	00:00:00.028845	00:00:00.015098	00:00:00.018718
Μέσος όρος	00:00:00.319770	00:00:00.165523	00:00:00.134972

Ερώτημα 4Γ: Σχολιασμός αποτελεσμάτων

Write Consistency - Mean Duration



Read Consistency - Mean Duration



Θεωρητικό υπόβαθρο για το CAP και τη Cassandra:

Το θεώρημα CAP (Consistency Availability Partition tolerance) αναφέρει ότι δεν είναι δυνατό να έχουμε ένα κατανεμημένο σύστημα βάσης δεδομένων που εγγυάται και τις τρεις ιδιότητες, τα αρχικά των οποίων αποτελούν το όνομα το θεωρήματος. Μόνο 2 από τις τρεις μπορούν να συντρέχουν.

Πιο συγκεκριμένα, όταν επιλέγεται και παραμετροποιείται ένα κατανεμημένο σύστημα βάσης δεδομένων γίνονται επιλογές για tradeoff μεταξύ υψηλού consistency και availability και χαμηλού partition tolerance, ή υψηλού consistency και partition tolerance και χαμηλού availability ή υψηλού availability και partition tolerance και χαμηλού consistency. Οι επιλογή που κάνει ο σχεδιαστής είναι με σκοπό να βρει το κατάλληλο tradeoff που πληρεί καλύτερα τις ανάγκες της εφαρμογής που θέλει να υλοποιήσει.

Η Cassandra είναι ένα παράδειγμα κατανεμημένης βάσης δεδομένων, που χρησιμοποιείται συνήθως για εφαρμογές που χρειάζονται υψηλό availability και partition tolerance. Παρόλα αυτά, το επίπεδο του consistency (συνέπεια) στη Cassandra μπορεί να ρυθμιστεί, βέβαια το tradeoff είναι ότι για μεγαλύτερο consistency καταλήγουμε με μικρότερο availability και partition tolerance.

Η Cassandra επιτυγχάνει μεγαλύτερη συνέπεια κατά απαίτηση μέσω της αύξησης απαιτήσεων αριθμού απαντήσεων από nodes που έχουν / αναλαμβάνουν replicas για τα δεδομένα που γίνονται read / written. Οπότε ο αριθμός απαντήσεων από nodes με replicas εξαρτάται από το consistency level και του replication factor.

Στη Cassandra οι ρυθμίσεις της συνέπειας είναι πολυάριθμες, αλλά αυτές που ζητήθηκαν να δοκιμαστούν είναι οι ONE, QUORUM και ALL. Η ρύθμιση ONE απαιτεί μόνο μια απάντηση από node με replica για να γίνει η εγγραφή ή ανάγνωση. Η ρύθμιση QUORUM απαιτεί αριθμό απαντήσεων ίσο με το μισό συν 1 του αριθμού των nodes με replicas (δηλαδή του replication factor). Τέλος η ρύθμιση ALL απαιτεί απαντήσεις από όλα τα nodes με replicas. Άρα το ONE δίνει μικρό επίπεδο συνέπειας, το QUORUM αποτελεί μια ενδιάμεση προσέγγιση και το ALL δίνει υψηλό επίπεδο συνέπειας.

Σημειώνω επίσης πως το write consistency αναφέρεται στη συνέπεια σε πράξης εγγραφής (INSERT εντολές στη περίπτωση αυτή), άρα απαιτούνται απαντήσεις επιτυχούς εγγραφής από τα nodes με replicas. Το read consistency αναφέρεται σε πράξης ανάγνωσης (SELECT εντολές στη περίπτωση αυτή), άρα απαιτούνται απαντήσεις της ανάγνωσης από τα nodes με replicas, όπου πλέον ο coordinator εμφανίζει στο χρήστη τη πιο πρόσφατη ανανεωμένη απάντηση που έλαβε από τα nodes.

Επεξήγηση μετρήσεων:

Σχετικά με το γράφημα του write consistency, παρατηρώ ότι ο μεγαλύτερος μέσος χρόνος ήταν για το consistency level ONE, έπειτα για το ALL και τέλος για το QUORUM. Σχετικά με τα αποτελέσματα αυτά, αναφέρω ότι η αναμενόμενη σειρά ήταν ο περισσότερος μέσος χρόνος να ήταν για το ALL, έπειτα για το QUORUM και τέλος για το ONE. Αυτό συμβαίνει επειδή ο coordinator για μεγαλύτερα consistency levels περιμένει περισσότερες απαντήσεις από τα nodes που έχουν replicas των δεδομένων, με αποτέλεσμα κάθε πράξη να απαιτήσει περισσότερο χρόνο, εφόσον οι περισσότερες απαντήσεις έχουν περισσότερο ενδεχόμενο

χρόνο να καθυστερήσουν και ο coordinator “περιμένει” παραπάνω για τη μέση εντολή. Άρα υποθέτω πως ο λόγος που το ONE άργησε παραπάνω σε αυτή τη περίπτωση από τα άλλα είναι λόγω πιθανού συνδυασμού υψηλού φόρτου δικτύου, πιθανότερα υψηλού φόρτου στο microcluster που οδήγησε σε εντολές να καθυστερήσουν παραπάνω είτε τυχαιότητας που προκύπτει λόγω του μικρότερου συνολικού όγκου δεδομένων που αναγκάστηκα να χρησιμοποιήσω. Επίσης αναφέρω πως αύξηση τιμής στη ρύθμιση ‘write_request_timeout_in_ms’ που αναγκάστηκα να χρησιμοποιήσω μπορεί να έπαιξε ρόλο, εφόσον τα requests για τα οποία καθυστερούν τα nodes υπόκεινται σε τυχαιότητα. Παρόλα αυτά, ήταν αναμενόμενο το QUORUM να χρειαστεί λιγότερο μέσο χρόνο από το ALL, όπως έδειξαν οι ρυθμίσεις. Πάντως η κύρια outlier τιμή που ‘αυξάνει’ το μέσο όρο του consistency ONE είναι η τιμή του insert για το πίνακα recipes_by_tag.

Σχετικά με το γράφημα του read consistency, παρατηρώ ότι ο μεγαλύτερος μέσος χρόνος ήταν για το consistency level ALL, έπειτα για το QUORUM, έπειτα για το ONE. Αυτή ήταν και η αναμενόμενη σειρά χρονικών απαιτήσεων που περίμενα, για τους ίδιους λόγους που εξηγήθηκαν και παραπάνω, με μόνη διαφορά ότι πλέον ο coordinator περιμένει πλέον για απαντήσεις από ένα ή πολλά nodes με replicas των δεδομένων με σκοπό να επιλέξει τα πιο πρόσφατα.

Βιβλιογραφία

1. Cassandra The Definitive Guide DISTRIBUTED DATA AT WEB SCALE - Jeff Carpenter & Eben Hewitt (2nd Edition) (Appendix 1 στο eclass)
2. A Big Data Modeling Methodology for Apache Cassandra - Artem Chebotko, Andrey Kashlev, Shiyong Lu (Appendix 2 στο eclass)
3. Οι διαφάνειες του μαθήματος
4. <https://javahowtos.com/guides/446-create-cassandra-cluster-with-docker-compose.html>
5. <https://cloudinfrastructureservices.co.uk/how-to-setup-cassandra-docker-container-using-docker-compose/>
6. <https://docs.datastax.com>
7. <https://stackoverflow.com> (για troubleshooting από errors που προέκυπταν)