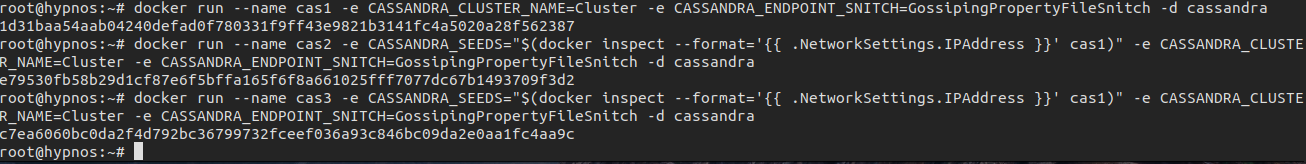
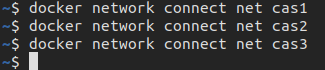
**Task 6 - Реплікация у Cassandra**

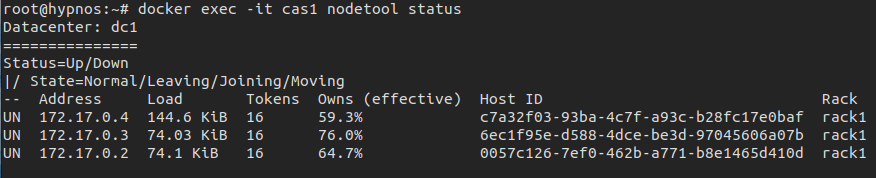
1. Сконфігурувати кластер з 3-х нод



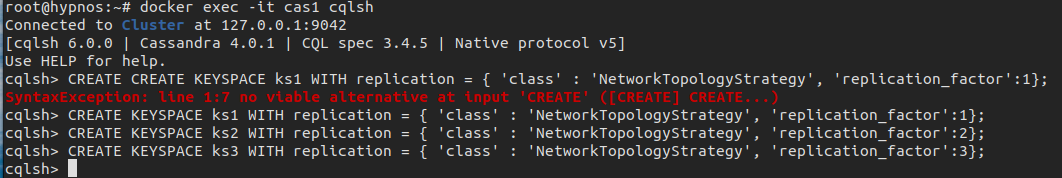


1. Перевірити правильність конфігурації за допомогою

*nodetool status*



1. Викоритовуючи *cqlsh*, створити три *Keyspace* з replication factor 1, 2, 3

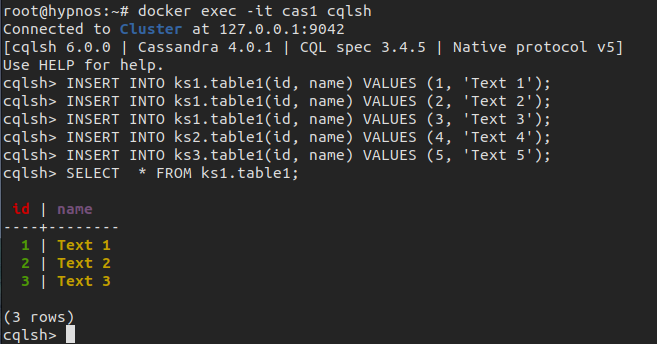


1. В кожному з кейспейсів створити таблиці

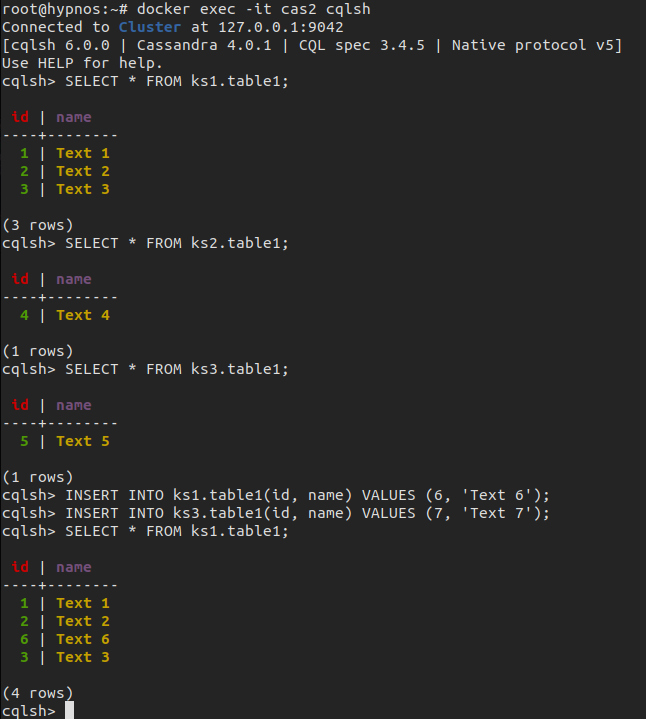


1. Спробуйте писати і читати на / та з різних нод.

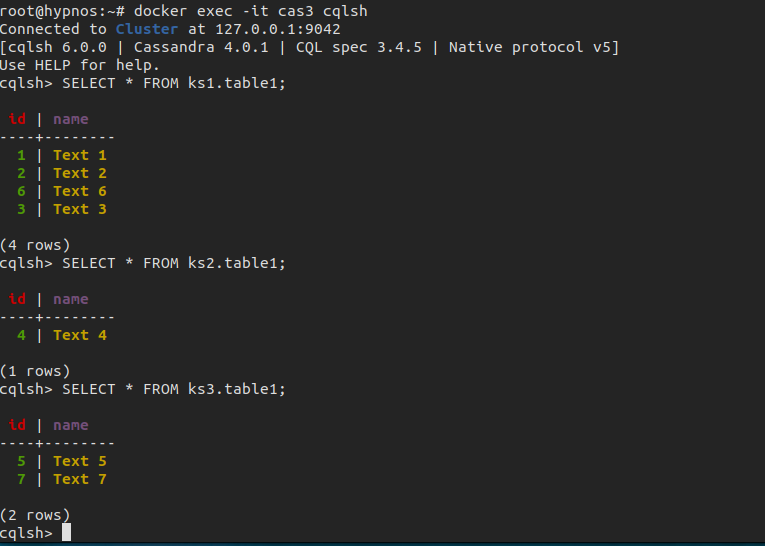
Node 1:



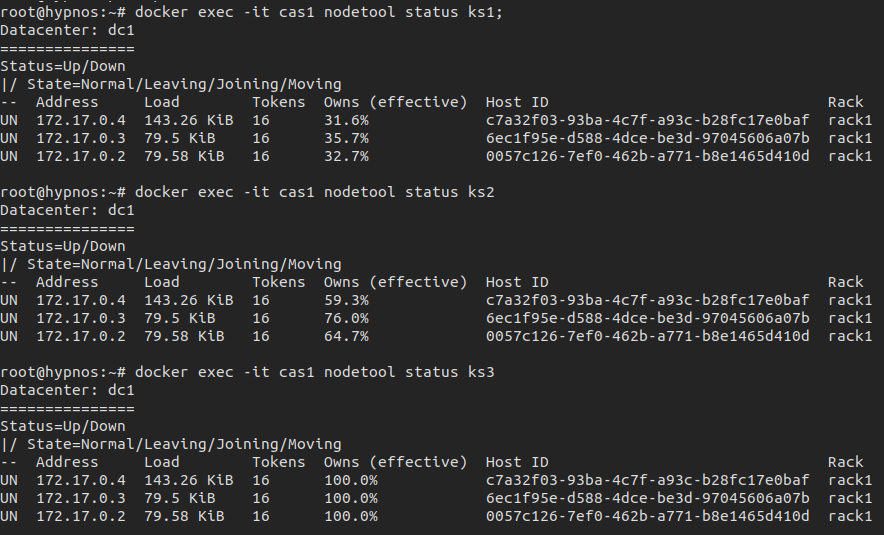
Node 2:



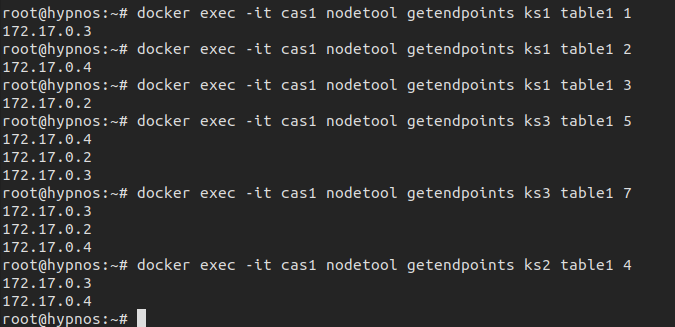
Node 3:



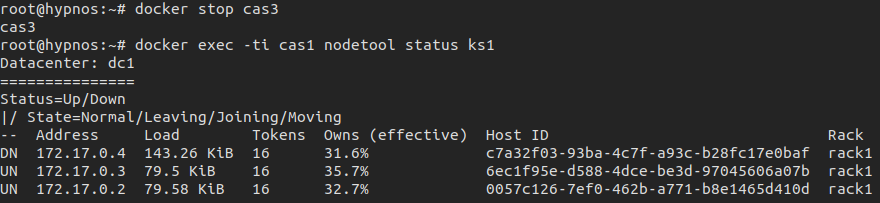
1. Вставте дані в створені таблиці і подивіться на їх розподіл по вузлах кластера (для кожного з кейспесов - *nodetool status*)

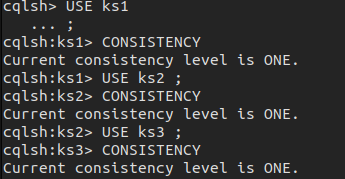


1. Для якогось запису з кожного з кейспейсу виведіть ноди на яких зберігаються дані



1. Відключиіть одну з нод. Для кожного з кейспейсів визначить з якими рівнями *consistency* можемо читати та писати, і які з них забезпечують *strong consistency*





R + W > N

ks1: N = 1, R = 1, W = 1 - strong consistency

ks2: N = 2, R = 1, W = 1 - weak consistency

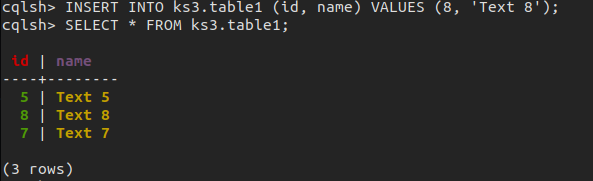
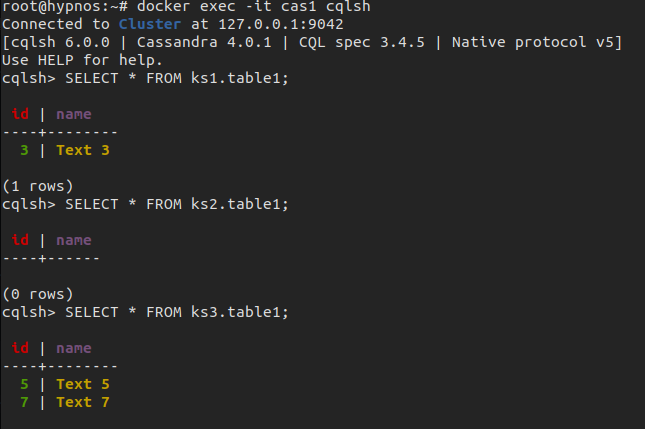
ks3: N = 2, R = 1, W = 1 - weak consistency

1. Зробить так щоб три ноди працювали, але не бачили одна одну по мережі (відключити зв'язок між ними)

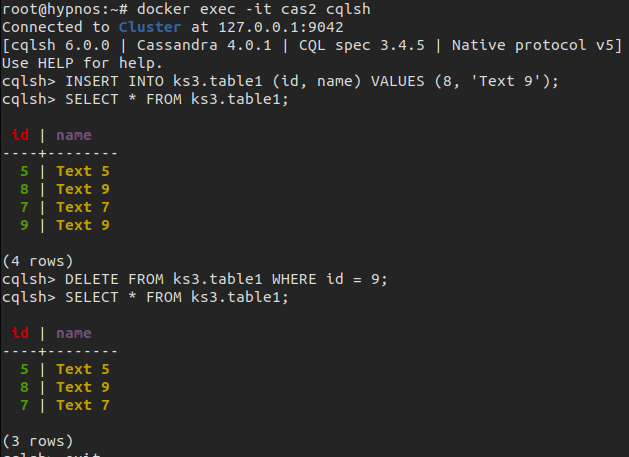


1. Для кейспейсу з *replication factor* 3 задайте рівень consistency рівним 1. Виконайте запис одного й того самого значення, з однаковим primary key, але різними іншими значенням на кожну з нод (тобто створіть конфлікт)

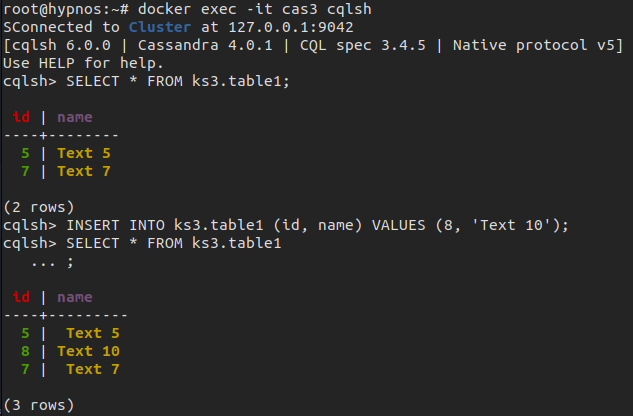
Node 1: (id 8, text 8)



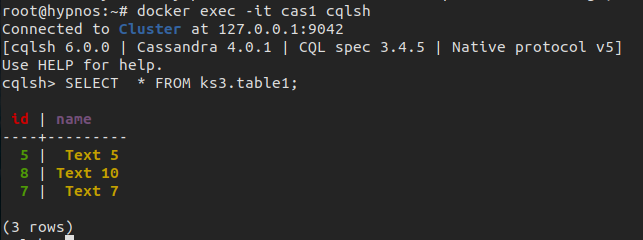
Node 2: (id: 8, text 9)



Node 3:



1. Об’єднайте ноди в кластер і визначте яке значення було прийнято кластером та за яким принципом



Було обрано найновіше значення з ноди 3.

1. Перевірте поведінку *lightweight transactions* для попередніх пунктів у розділеному на три частини кластері

