### fIndexes

Help with query performance

Without secondary indexes, clients must either skip the where clause, which forces cassandra to traverse all nodes, or include a where clause containing a partition key.

Secondary index allow you to query based on more than just partition key and clustering columns.

create index users\_company ON users(company)l

Indexes can also be created on collections.

alter table users add tags set<varchar>

// naming the index is not required

create index on users(tags);

### Querying indexed collections

select \* from users where tags CONTAINS ‘java’

Two forms

Create index on a collection column

allows use of CONTAINS in select statements

Works for all three collection types (set, list, map)

For map, CONTAINS matches the values of a map

If you want to match on the keys of a map, use the KEYS keyword

create index on foo (KEYS(bar));

and when querying, use the CONTAINS KEY construct

select \* from doo where CONTAINS KEY 123;

## When to avoid secondary indexes

Building indexes over columns with lots of distinct values (high cardinality)

High or very low cardinality columns

Batching

Compare and set operations using transactions

## Primary vs Secondary indexes

primary indexes are stored globally

secondary indexes are stored locally

When querying by primary key, every node knows which machine to go to for the record.

When querying by secondary index, every machine in the cluster must be read from.

Denormalized index

## Reference