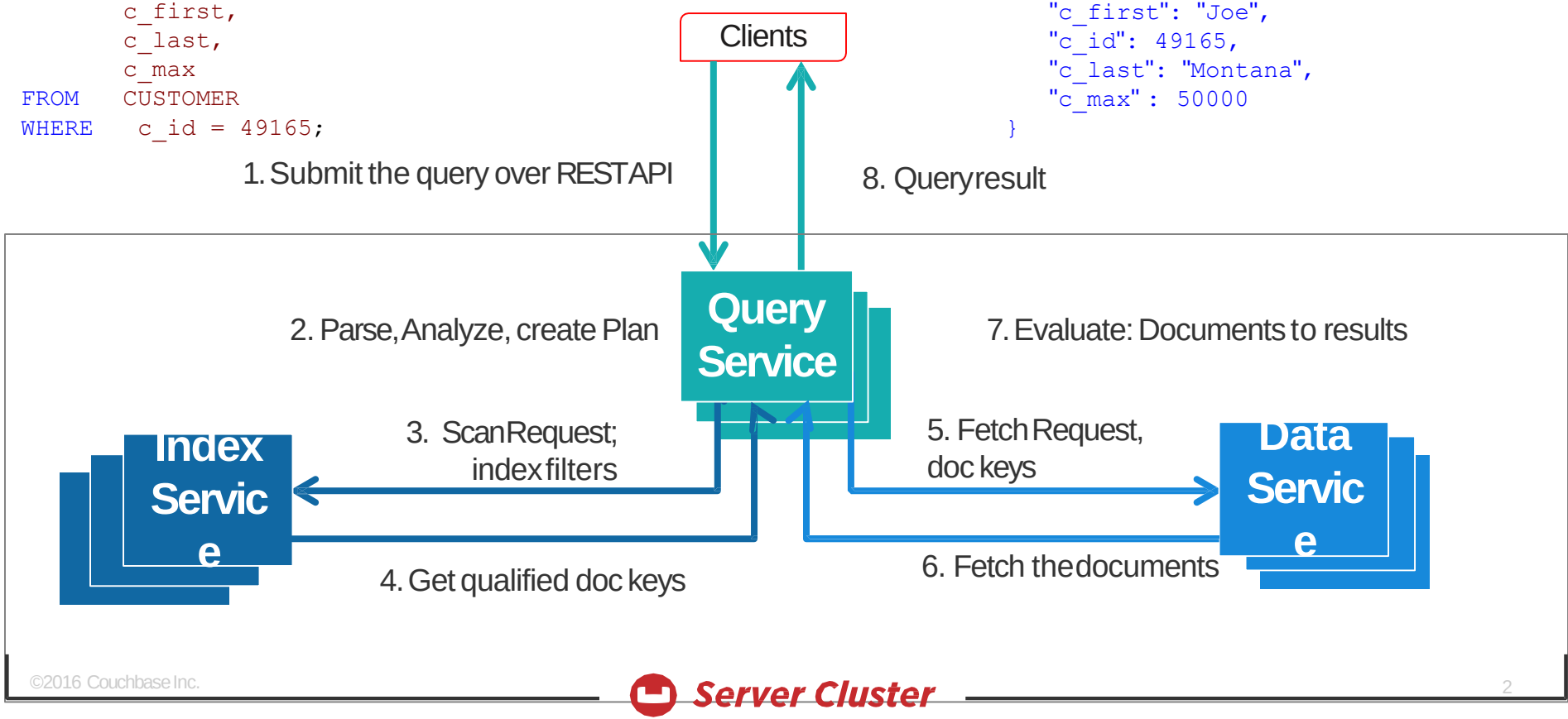
QUERY EXECUTION

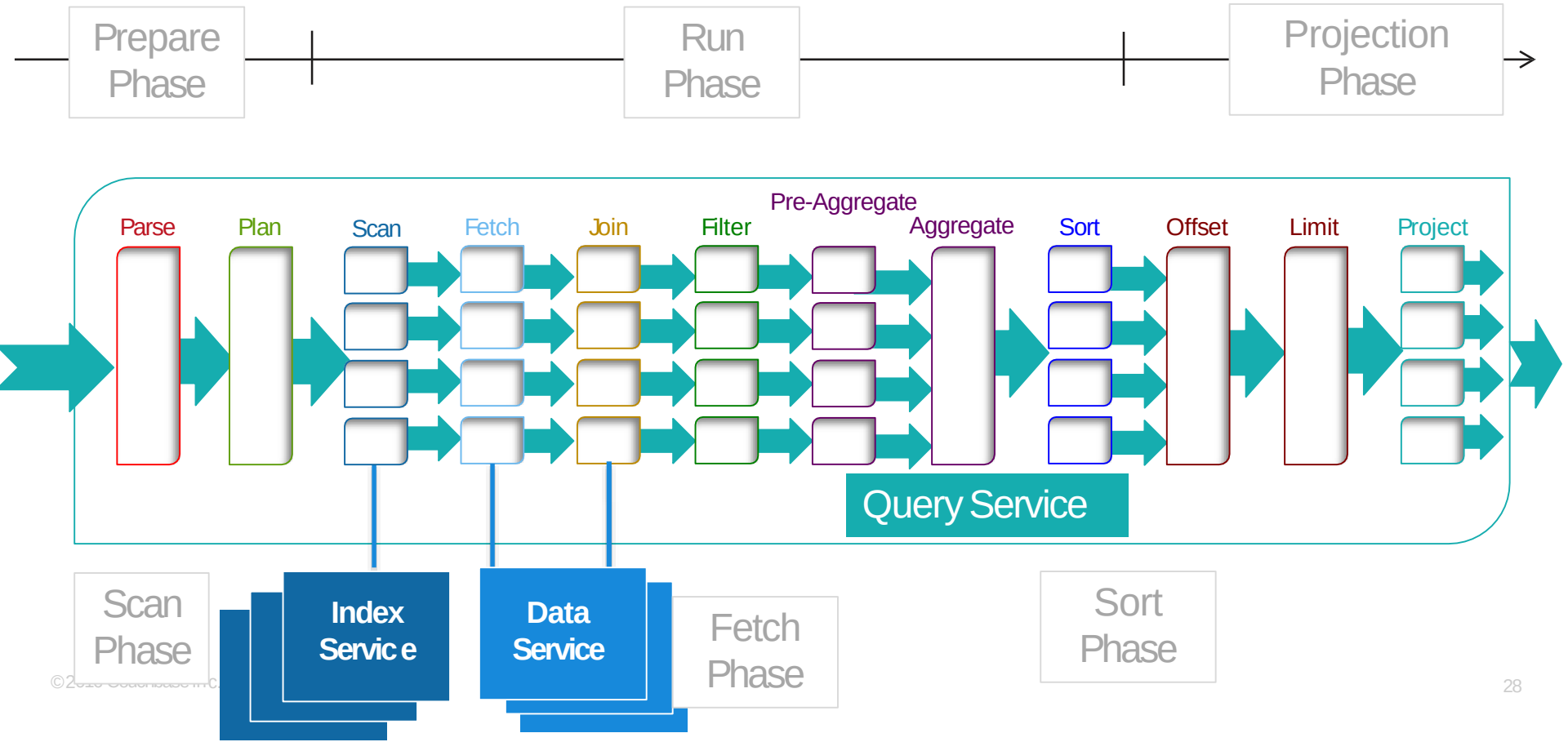
N1QL: Query Execution Flow

```
SELECT c_id,
       c_first,
       c_last,
       c_max
FROM   CUSTOMER
WHERE   c_id = 49165;
```

```
{
   "c_first": "Joe",
   "c_id": 49165,
   "c_last": "Montana",
   "c_max" : 50000
}
```



Inside a Query Service



Index Overview: PrimaryIndex

- Primary Index
  - `CREATE PRIMARY INDEX` `` `beer-sample` ``;
- Document key is unique for the bucket.
- Primary index is used when no other qualifying index is available or when no predicate is given in the query.
- PrimaryScan is equivalent of full table scan

Index Overview: Secondary Index

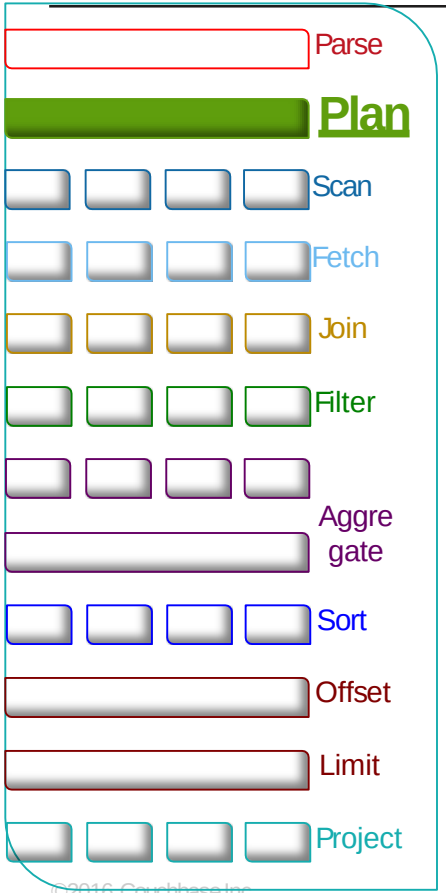
Document key: "Pale Ale.A1"

Secondary Index can be created on any combination of attribute names.

```
create index idxbeerabv on`beer-
sample`(abv, state);
create index  idxtypeabvstate on
`beer-sample`(abv, state) where type =
'beer';
```

- Need to have matching indices with right key-ordering

Query Execution: Plan




```
create index idxbeerabv on
                    `beer-sample`(abv, state);

EXPLAIN SELECT count(*)
FROM   `beer-sample`
WHERE  abv > 5.0 AND state = 'California';

select *
from system:indexes
where keyspace_id = 'beer-sample';
```

- Explain provides the JSON representation of the query plan
- Focus on the index selection and the predicates pushed down

Explain

The EXPLAIN statement when used before any N1QL statement, provides information about the execution plan for the statement.

```
EXPLAIN SELECT title, genre, runtime FROM catalog.details
ORDER BY title
```

Explain….


```
"results": [
   {
      "#operator": "Sequence",
      "~children": [
          {
             "#operator": "Sequence",
             "~children": [
                 {
                    "#operator": "PrimaryScan",
                    "index": "#primary",
                    "keyspace": "catalog",
                    "namespace": "default"
                 },
                 {
                    "#operator": "Parallel",
                    "~child": {
                         "#operator": "Sequence",
                         "~children": [
                              {
                                 "#operator": "Fetch",
                                 "keyspace": "catalog",
                                 "namespace": "default",
                                 "projection": "`details`
                              },
                              {
                                 "#operator": "InitialProject",
                                 "result_terms": [
                                     {
                                       "expr": "(`details`.`title`)"
                                     },
                                     {
                                       "expr": "(`details`.`genre`)"
                                     },
                                     {
                                       "expr": "(`details`.`runtime`)"
                                     }
                                     ]
                                 }
                             ]
                 }
             },
             {
                 "#operator": "Order",
                 "sort_terms": [
                     {
                         "expr": "(`details`.`title`)"
                     }
                 ]
             },
             {
                 "#operator": "Parallel",
                 "~child": {
                     "#operator": "FinalProject"
                 }
      }
   ]
```

Plan Directives: USE INDEX

```
create index idxbeerabv on `beer-sample`(abv);
create index idxbeercat on `beer-sample`(category);
select *
from  `beer-sample` USE INDEX (idxbeerabv)
where abv > 15 and category like '%North%';

select *
from  `beer-sample` USE INDEX (idxbeerabv, idxbeercat)
where abv > 15 and category like '%North%';
```

- USE INDEX suggests one or more appropriate indices
- Optimizer validates the indices. Primary Index is the backup index.
- Provides stability to the optimizer plan.

Covering Index

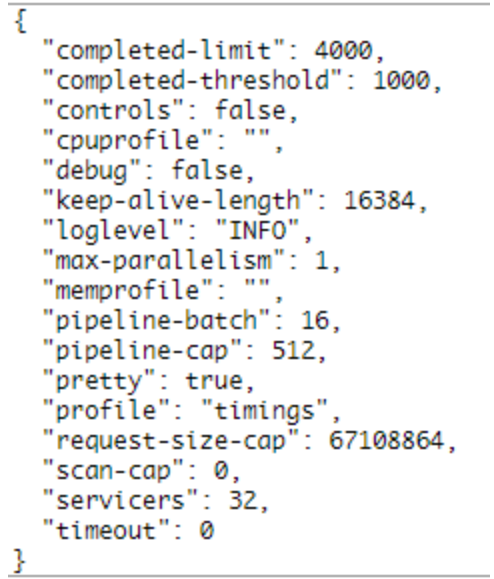
N1QL: Example

```
create index idxstatecity on `beer-
sample`(state,city) using gsi;

EXPLAIN
SELECT city
FROM`beer-sample`
WHERE state = 'California';
```


```
{
 "requestID": "eef73760-d09e-48e0-a43a-c8da1e0be998
 "signature": "json",
 "results": [
   {
     "#operator": "Sequence",
     "~children": [
       {
         "#operator": "IndexScan",
          "covers": [
            "cover((meta(`beer-sample`).`id`))",
            "cover((`beer-sample`.`state`))",
            "cover((`beer-sample`.`city`))"
         ],
         "index": "idxstatecity",
         "keyspace": "beer-sample",
         "namespace": "default",
         "spans": [
           {

                    ]
```

Profiling

http://localhost:8093/admin/settings, POST

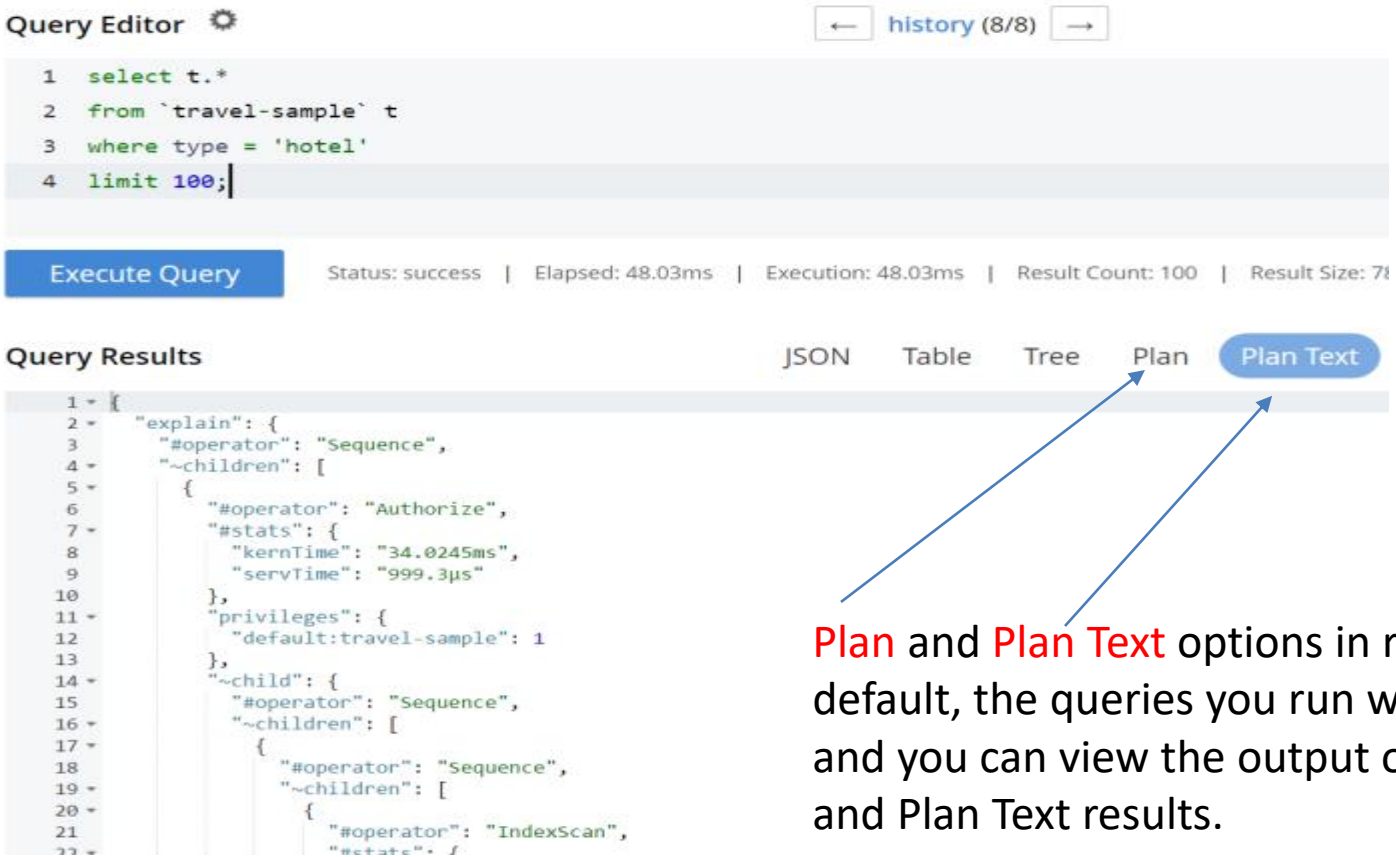


- We can collect execution timings and document processed on a per operator basis
- If the functionality is turned on, timings are reported
  - with the metrics at the end of execution
  - in system:active_requests
  - in system:completed_requests
- Profiling is turned on
  - at the request level via the “profile” REST API parameter,
  - EG from cbq:
    - \set –profile timings;
  - at the node level via the “profile” command line parameter or admin settings REST API parameter
  - takes 3 values, “off”, “phases”, “timings”
    - “phases” supplies total times for each operator class
    - “timings” supplies detailed information for each operator

Profiling

```
cbq> select * from `travel-sample` where source-airport is not missing;
...
"executionTimings": {
                "~children": [
                    {
                        "#operator": "IndexScan2",
                           "#stats": {
                                "#itemsOut": 24024,
                                "#phaseSwitches": 96099,
                                "execTime": "55.370283ms",
                                "kernTime": "5.397199311s"
                           },
                        "index": "def_sourceairport",
                        "index_id": "29702e564c9d2ca4",
                        "index_projection": {
                            "primary_key": true
                        },
                        "keyspace": "travel-sample",
                        "namespace": "default",
                        "spans": [
                            {
                                "exact": true,
                                "range": [
                                    {
                                        "inclusion": 1,
                                        "low": "null"
```

Query plan visualization



Plan and Plan Text options in results. By default, the queries you run will be EXPLAINed, and you can view the output of that in the Plan and Plan Text results.

Lab : Tuning Query – (Explain to determine query plan)