Couchbase

Using the Couchbase C/C++ Client Library

Workshop Day 2

https://github.com/dmaier-couchbase/cb-workshop-cpp

http://www.bit.ly/amadeus-cpp

Before we begin

- Make sure that Couchbase Server is installed on the Dev Machine!

Document Modelling Basics

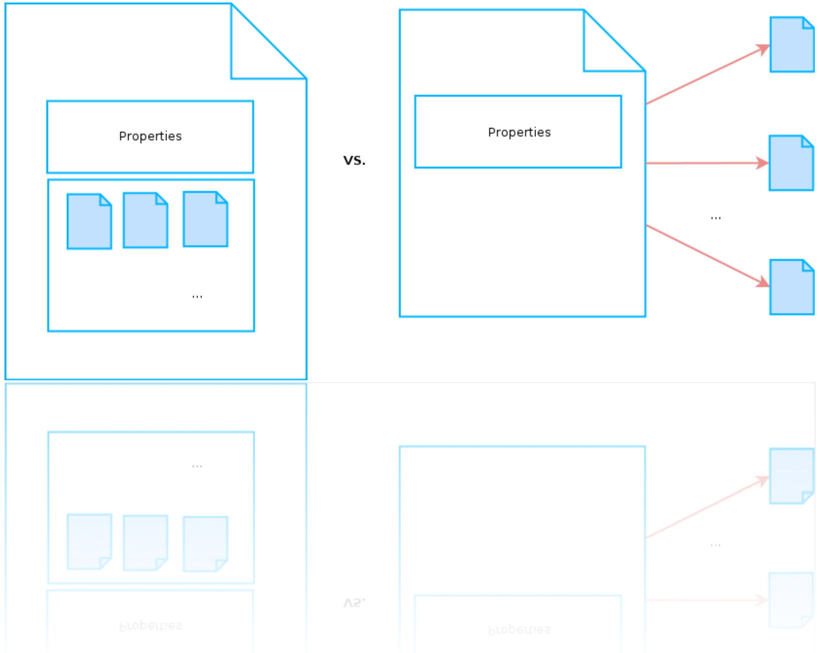
JSON

- Java Script Object Notation
  - Meta data
  - Document Value

```
"meta" :
{
  "id" : "person::david",
  "rev" : "1-0002bce0000000000",
  "flags" : 0,
  "expiration":0,
  "type":"json"
}

"doc" :
{
  "type" : "person",
  "uid":"david",
  "firstname":"David",
  "lastname":"Maier",
  "birthday": 330004800000,
  "email":"david.maier@couchbase.com"
}
```

Normalization vs. De-Normalization

- Normalized
  - Uses key references for 1-many relationships
  - Reduces data duplicates
  - Smaller document size
- De-Normalized
  - Uses nested documents
  - Aggregated view of data
  - Allows atomic access
  - No client side joins



Normalization vs. De-Normalization

DE-NORMALIZED	NORMALIZED

```
{
  "type" : "organization",
  "oid" : "CB",
  "name" : "Couchbase",
  "street" : "2440 West El Camino Real Suite 101",
  "city" : "Mountain View",
  "state" : "California"
  "employees" :
   [
     {
       "uid":"david",
       "firstname":"David",
       "lastname":"Maier",
       "birthday": 1402920000000,
       "email":"david.maier@couchbase.com"
     },
     ...
   ]
}
```

```
{
  "type" : "organization",
  "name" : "Couchbase",
  "street" : "2440 West El Camino Real Suite 101",
  "city" : "Mountain View",
  "state" : "California"
  "employees" : ["person::david", "person::perry", "person::dipti", ... ]
}
```

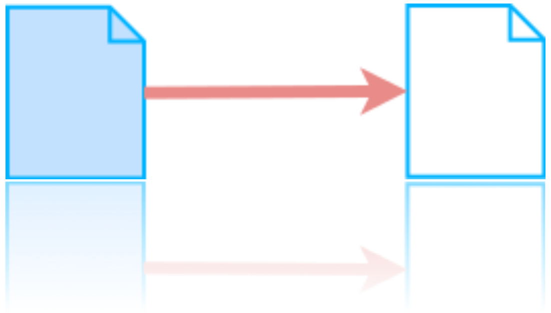
Atomic Counters

- Similar to sequences / auto-incrementing columns from the relational world
- Initialize and increment a counter value
- Use the counter as part of the key

```
id = client.incr("count::person");


client.add("person::" + id, doc);
```

Reference Documents for Lookups

- Second document which references the primary one
- Needs to be maintained by the application



```
"email::david.maier@couchbase.com" : { "ref" : "person::david" }
```

Managing Connections

Exercise 7

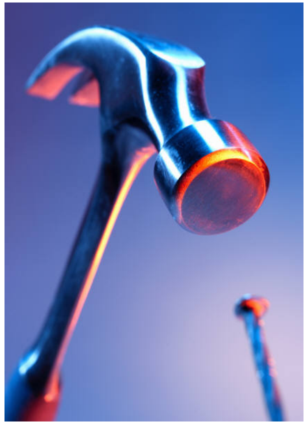
libcouchbase Installation

Perform the following steps in order to install libcouchbase

- Perl needs to be installed

  http://developer.couchbase.com/documentation/server/current/sdk/c/start-using-sdk.html

  ```
  su root
  rpm -iv couchbase-release-1.0-2-x86_64.rpm
  yum install libcouchbase-devel libcouchbase2-bin
  ```



- The described setup procedure adds the the Couchbase package repository (/etc/yum.repos.d) and then installs the packages 'libcouchbase2-bin' and 'libcouchbase-devel'.

Get the Workshop Sources

Perform the following steps in order to check out the latest source code

- New installation

```
git clone https://github.com/dmaier-couchbase/cb-workshop-cpp.git
```

- Preinstalled workshop machine

```
cd ~/Git/cb-workshop-cpp
git rebase
```

Before we begin

Open the documentation for libcouchbase!

- http://developer.couchbase.com/documentation/server/4.5/sdk/c/start-using-sdk.html

- Provided helper classes

    CouchbaseDocument
    CBCookie*
    CBQStringConvert

Connecting to Couchbase

Implement the following methods in CBDataSource:

- void Connect(QString connectionString, QString password);

Implement the following methods in CBDataSourceFactory:

- static void Create(QString connectionString, QString password);

Test your implementation by executing:

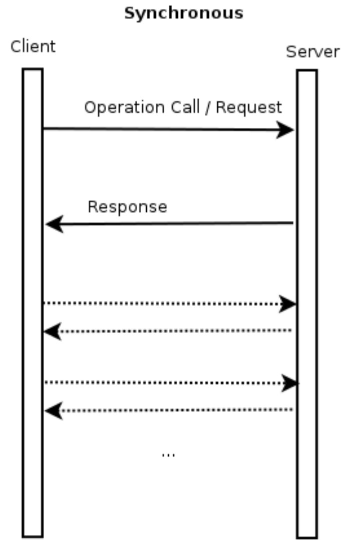
```
DemoCouchbaseConnect connectDemo;
connectDemo.test();
```

Understanding Non-Blocking I/O

in libcouchbase

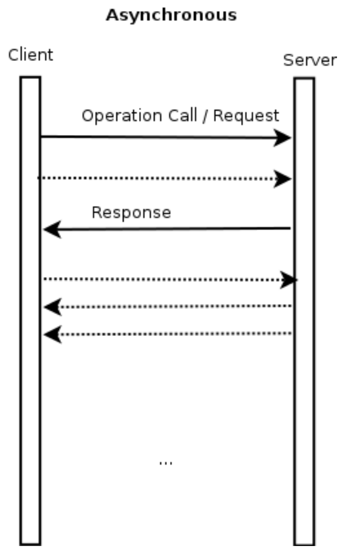
Blocking

- Libcouchbase is designed to use non-blocking I/O
  - Scheduled operations
- But lcb_wait() blocks by default
  - Waits for pending requests
  - Used for synchronous operation execution
- Callback functions are used
  - e.g. storage_callback



Non-Blocking

- External event loop integration
  - Provides mechanism to execute a callback function when a specific event occurs
  - e.g. libevent
- Asynchronous operation execution
- No need for lcb_wait()



Working with Documents

Exercise 8 - 11

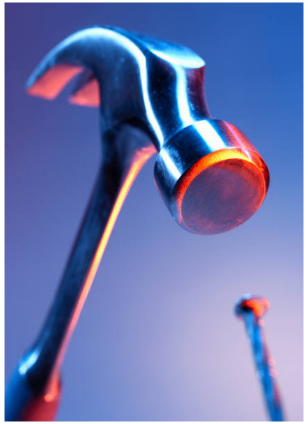
Get a Document

Make sure that the travel-sample data is installed!
Implement the following methods in CBDataSource:

- CouchbaseDocument Get (QString key);

Test your implementation by executing:

```
DemoCouchbaseGet getDemo;
getDemo.test();
```



Perform a Multi-Get

Make sure that the travel-sample data is installed!

Implement the following methods in CBDataSource:

- CouchbaseDocumentMap MultiGet(QStringList keys);

Test your implementation by executing:

```
DemoCouchbaseMultiGet multiGetDemo;
multiGetDemo.test();
```

Create/Update a Document

Implement the following methods in CBDataSource:

- bool Upsert(QString key, QString document)

Test your implementation by executing:

```
DemoCouchbaseUpsert upsertDemo;
upsertDemo.test();
```

Delete a Document

Implement the following methods in CBDataSource:

- bool Delete(QString key);

Test your implementation by executing:

```
DemoCouchbaseDelete deleteDemo;
deleteDemo.test();
```

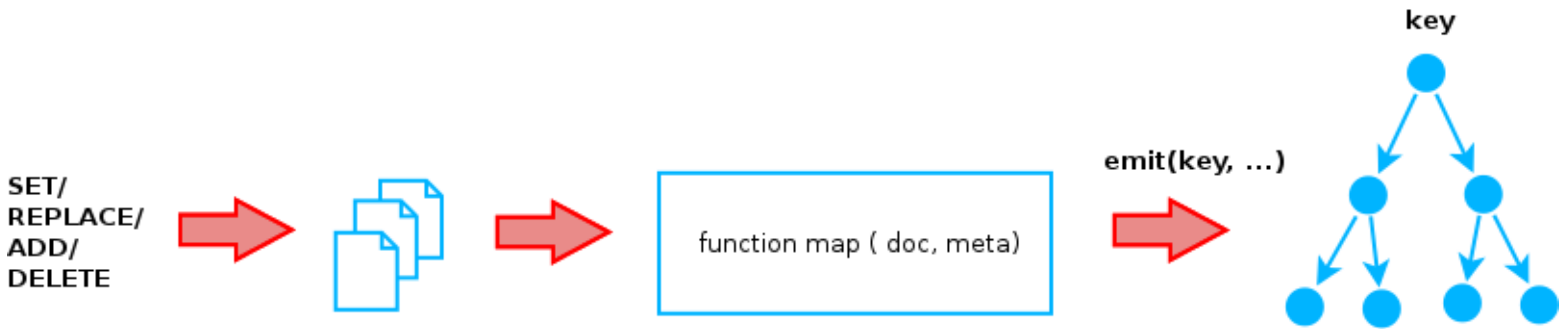
Querying via Views

Exercise 12

Views

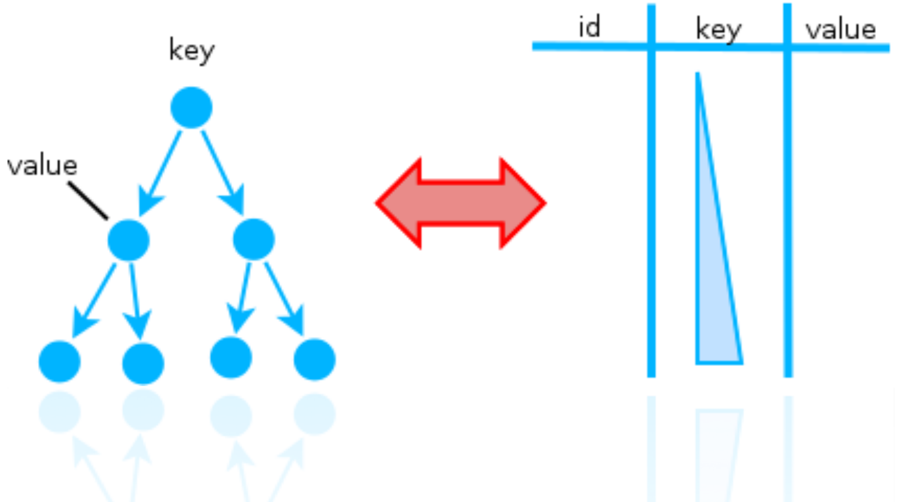
- Organized in Design Documents
- Incremental Map-Reduce
- Spread indexing load across nodes

Map	Reduce
Process, filter, map and emit a row	Aggregate mapped data Built in: _count, _sum, _stats



Views

- Multiple roles
  - A Primary Index to access all document id-s
  - A Secondary Index as an alternative access path
  - A View provides you an alternative view on your data



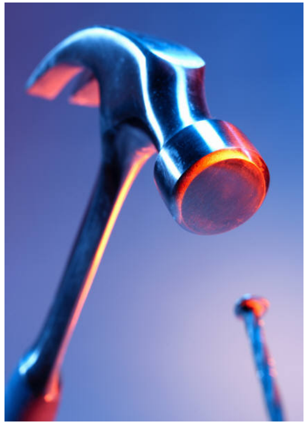
Query a View

Create the View 'airports/by_name' !
Implement the following methods in CBDataSource:

- CBQueryResult QueryView(QString designDocName,
  QString viewName, int limit=0, int skip=0);

Test your implementation by executing:

```
DemoCouchbaseView viewDemo;
viewDemo.test();
```



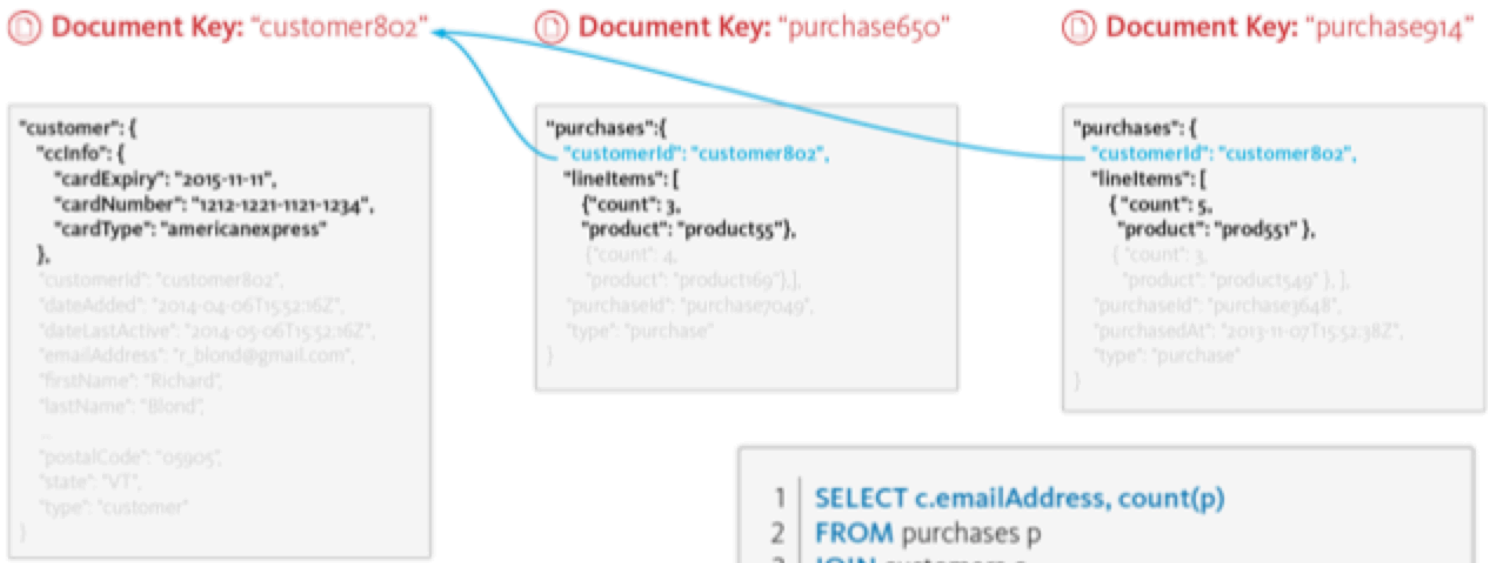
Querying via N1QL

Exercise 13

N1QL Introduction

- Next generation, NoSQL query language
- SQL-like
    - WHERE
    - LIKE
    - GROUP
    - JOINS
- Powerful Extensions for JSON and hierarchical data structures
    - NEST
    - UNNEST
- Multiple access paths
    - Views
    - Global Secondary Indexes
        - Memory Optimized Indexes

N1QL Introduction - Joins



Document Key: "customer802"

```
"customer": {
  "ccInfo": {
    "cardExpiry": "2015-11-11",
    "cardNumber": "1212-1221-1121-1234",
    "cardType": "americanexpress"
  },
  "customerId": "customer802",
  "dateAdded": "2014-04-06T15:52:16Z",
  "dateLastActive": "2014-05-06T15:52:16Z",
  "emailAddress": "r_blond@gmail.com",
  "firstName": "Richard",
  "lastName": "Blond",
  ...
  "postalCode": "05905",
  "state": "VT",
  "type": "customer"
}
```

Document Key: "purchase650"

```
"purchases":{
  "customerId": "customer802",
  "lineItems": [
    {"count": 3,
    "product": "product55"},
     {"count": 4,
     "product": "product169"},],
  "purchaseId": "purchase7049",
  "type": "purchase"
}
```

Document Key: "purchase914"

```
"purchases": {
  "customerId": "customer802",
  "lineItems": [
    { "count": 5,
     "product": "prod551" },
    { "count": 3,
      "product": "product549" }, ],
  "purchaseId": "purchase3648",
  "purchasedAt": "2013-11-07T15:52:38Z",
  "type": "purchase"
}
```

```
SELECT c.emailAddress, count(p)
FROM purchases p
JOIN customers c
ON KEYS (p.customerId)
GROUP BY c.emailAddress;
```

N1QL Query Examples

```
SELECT airportname FROM `travel-sample` WHERE
faa ='LAX'
```

```
SELECT airportname FROM `travel-sample` WHERE
faa ='LHR'
```

```
SELECT faa as fromAirport,geo FROM `travel-sample`
WHERE airportname = 'Los Angeles Intl' UNION
SELECT faa as toAirport,geo FROM `travel-sample`
WHERE airportname = 'Heathrow'
```

```
SELECT r.id, a.name, s.flight, s.utc, r.sourceairport,
r.destinationairport, r.equipment FROM `travel-sample` r
UNNEST r.schedule s JOIN `travel-sample` a ON KEYS
r.airlineid WHERE r.sourceairport='LHR' AND
r.destinationairport='LAX' AND s.day=6 ORDER BY a.name
```

```
SELECT airportname FROM `travel-sample` WHERE
airportname LIKE 'Los An%'
```

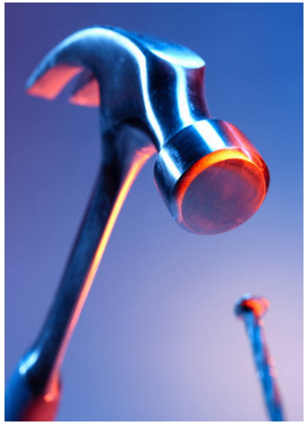
Query via N1QL

Make sure that at least a Primary Index is created!
Also Double check that the Secondary Index on ‘faa’ is there!
Implement the following methods in CBDataSource:

- CBN1qlResult QueryN1ql(QString query);

Test your implementation by executing:

```
DemoCouchbaseN1ql n1qlDemo;
n1qlDemo.test();
```



Error Handling and Logging

Handling errors

- Operations return lcb_error_t status code
- Check for
  - err == LCB_SUCCESS
- Error Codes
  - <libcouchbase/error.h>
- Examples
  - LCB_KEY_EEXISTS: Key already exists
  - LCB_KEY_ENOENT: Key does not already exist if replacing it
  - LCB_ETIMEDOUT: Transient error which indicates that something took too long
  - LCB_ETMPFAIL: Transient error which indicates that the server was too busy
  - LCB_AUTH_ERROR: Authentication error
  - LCB_BUCKET_ENOENT: Bucket does not exist

Logging

- LCB_LOGLEVEL environment variable
  - 1 – basic
  - 5 – verbose
- Programmatically
  - LCB_CNTL_CONLOGGER_LEVEL setting
  - console_log_level option in the connection string
- Log entry format

```
1ms [I0] {14780} [DEBUG] (lcbio_mgr - L:383) <localhost:11210> (HE=0xe56760)
Creating new connection because none are available in the pool
```

A Sample Application

Exercise 14

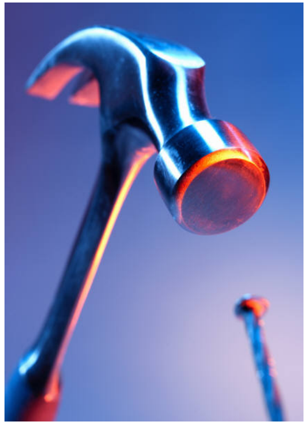
https://github.com/dmaier-couchbase/cb-workshop-cpp/tree/master/TravelAppSample

A Sample Application

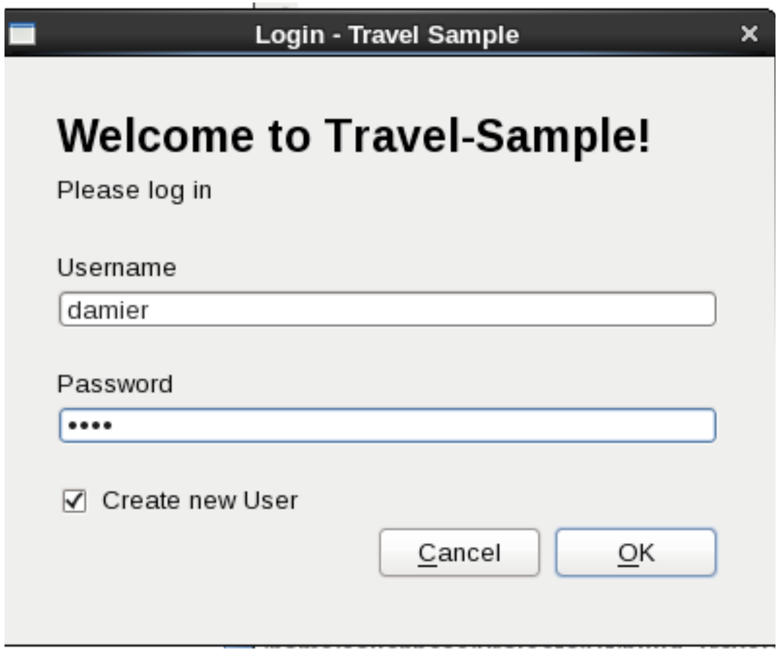
Inspect the full source code of the Travel-Sample application!

Run the Qt application!

- Search for a flight from ‘LAX’ to ‘LHR’



A Sample Application


Login - Travel Sample

Welcome to Travel-Sample!

Please log in

Username

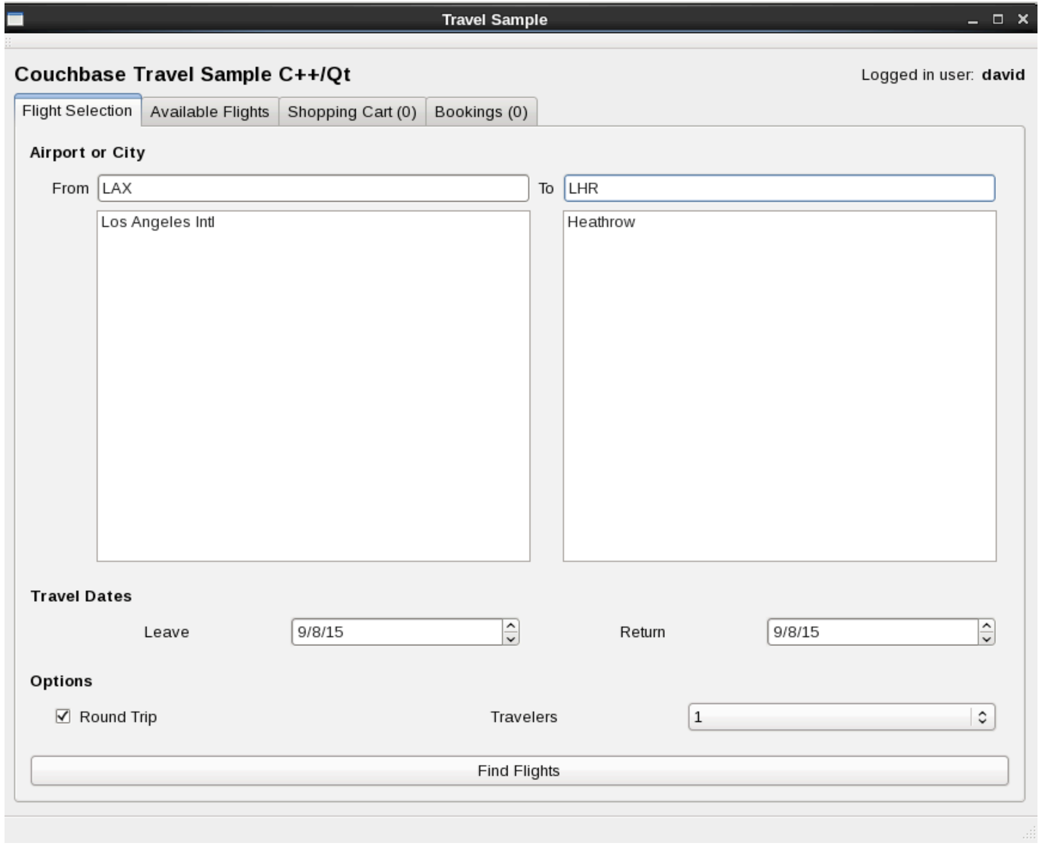
damier

Password

••••

✓ Create new User

A Sample Application



Travel Sample

Couchbase Travel Sample C++/Qt

Logged in user: **david**

Flight Selection | Available Flights | Shopping Cart (0) | Bookings (0)

Airport or City

From: LAX

To: LHR

Los Angeles Intl

Heathrow

Travel Dates

Leave: 9/8/15

Return: 9/8/15

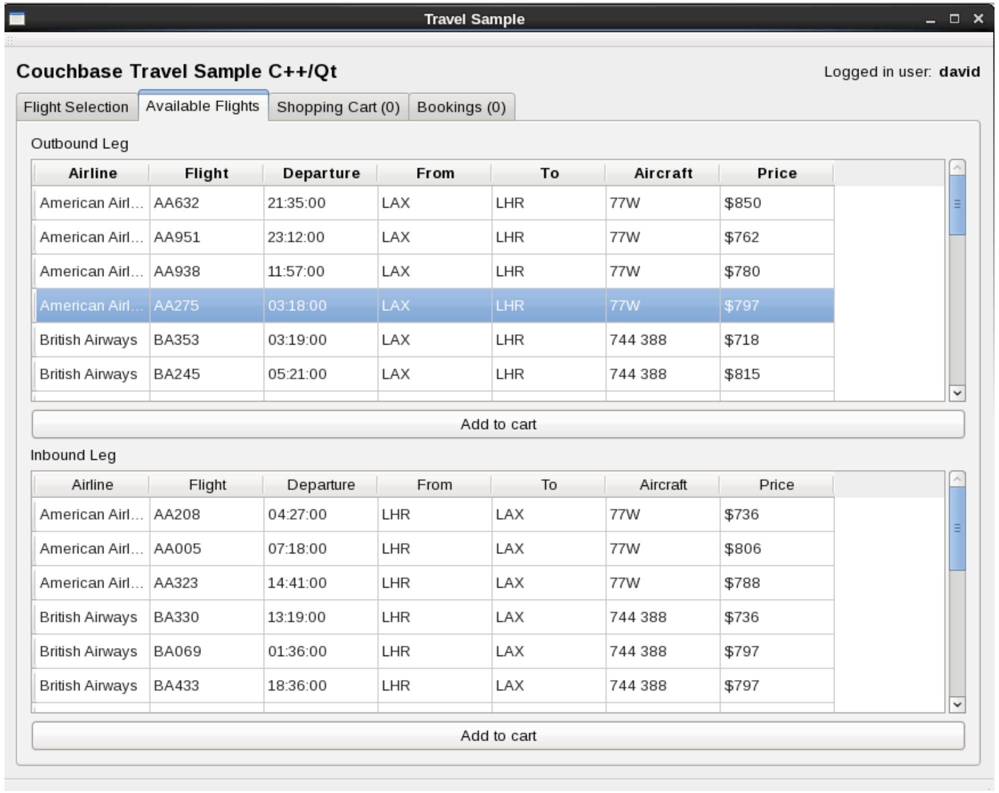
Options

☑ Round Trip

Travelers: 1

Find Flights

A Sample Application



Couchbase Travel Sample C++/Qt

Logged in user: **david**

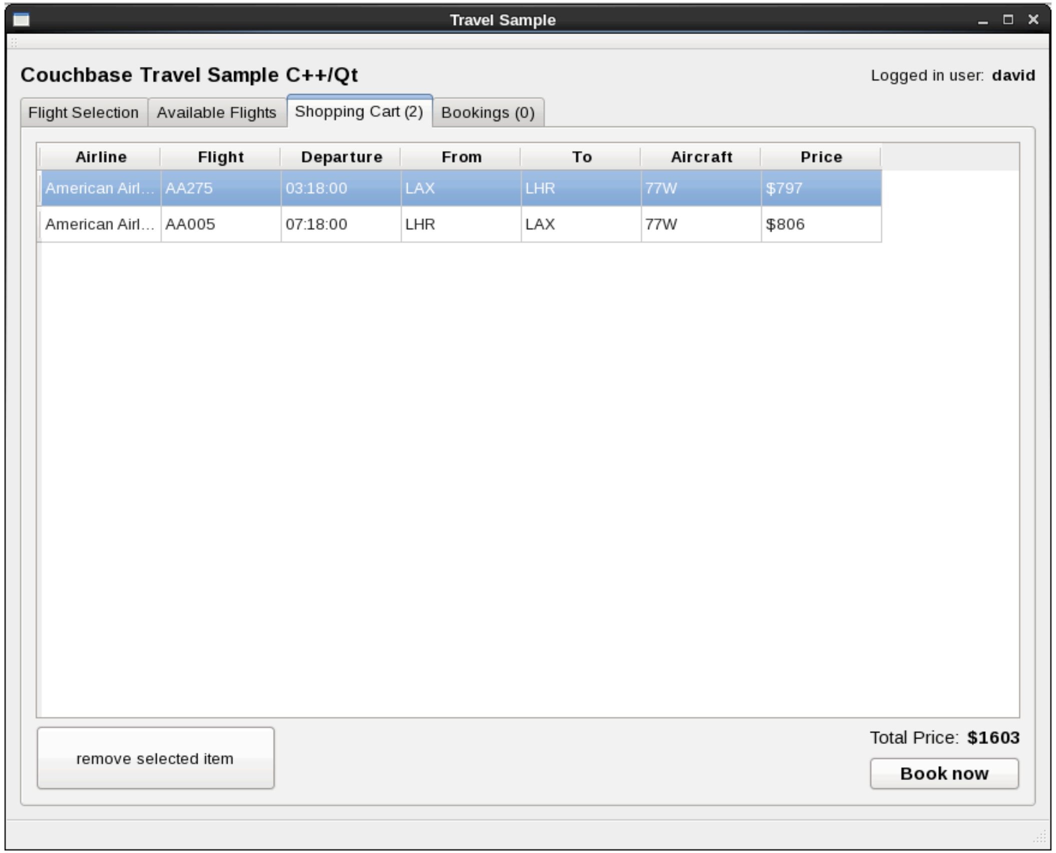
Outbound Leg

Airline	Flight	Departure	From	To	Aircraft	Price
American Airl…	AA632	21:35:00	LAX	LHR	77W	$850
American Airl…	AA951	23:12:00	LAX	LHR	77W	$762
American Airl…	AA938	11:57:00	LAX	LHR	77W	$780
American Airl…	AA275	03:18:00	LAX	LHR	77W	$797
British Airways	BA353	03:19:00	LAX	LHR	744 388	$718
British Airways	BA245	05:21:00	LAX	LHR	744 388	$815

Inbound Leg

Airline	Flight	Departure	From	To	Aircraft	Price
American Airl…	AA208	04:27:00	LHR	LAX	77W	$736
American Airl…	AA005	07:18:00	LHR	LAX	77W	$806
American Airl…	AA323	14:41:00	LHR	LAX	77W	$788
British Airways	BA330	13:19:00	LHR	LAX	744 388	$736
British Airways	BA069	01:36:00	LHR	LAX	744 388	$797
British Airways	BA433	18:36:00	LHR	LAX	744 388	$797

A Sample Application



Travel Sample

Couchbase Travel Sample C++/Qt

Logged in user: **david**

Airline	Flight	Departure	From	To	Aircraft	Price
American Airl…	AA275	03:18:00	LAX	LHR	77W	$797
American Airl…	AA005	07:18:00	LHR	LAX	77W	$806

Total Price: **$1603**

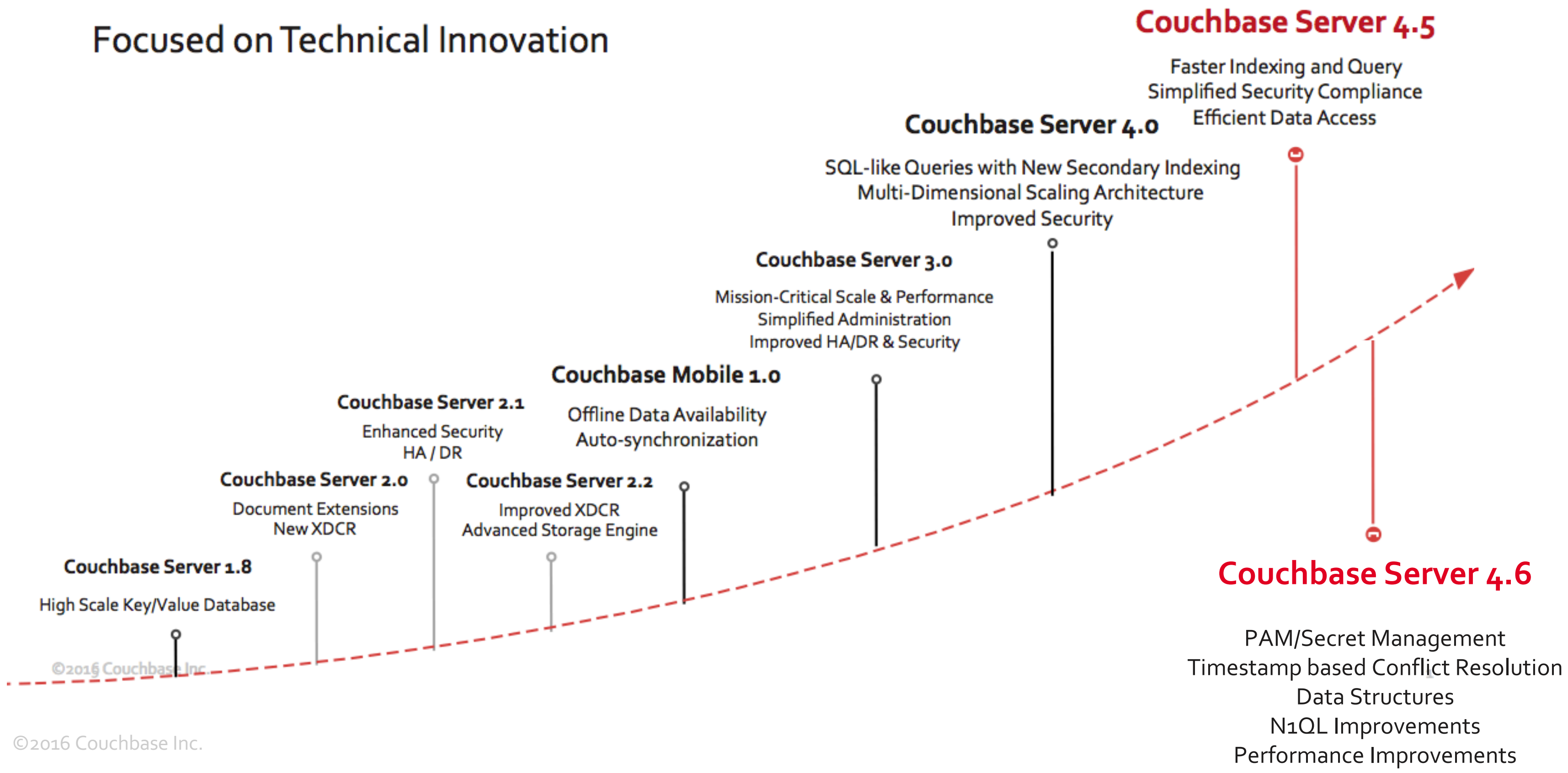
Q&A

http://docs.couchbase.com/developer/c-2.4/c-intro.html

Specific Use Case Presentation

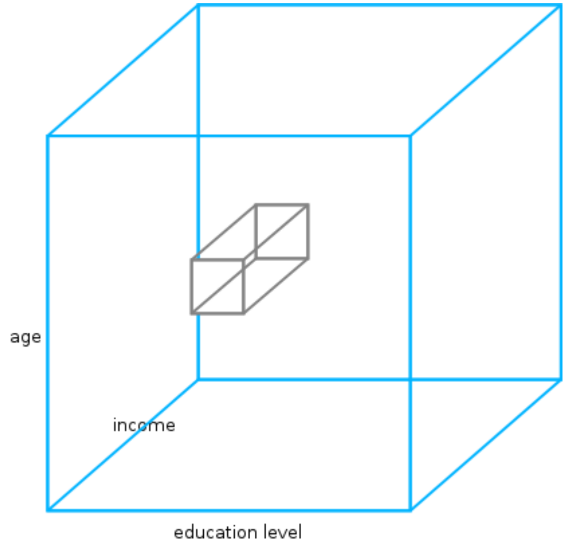
What’s new in 4.x?

Journey so far



Spatial Indexes

- Multi Dimensional Analysis
  - Not necessarily Geo-Data but any numeric data
  - Query within a Hyper-Cube
  - Map categories to numbers
- e.g.
  - Income, Age, Education level (Bsc = 4, Msc = 5)
  - Timestamp, Log-Level



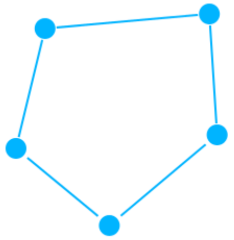
Spatial Indexes

- Geo-Data
  - GeoJSON: the “Open Standard”
  - More complex geometries stored as regions
  - Bounding-Box Queries
  - e.g. all buildings in San Francisco

```
{ "type": "Point", "coordinates": [100.0, 0.0] }
```

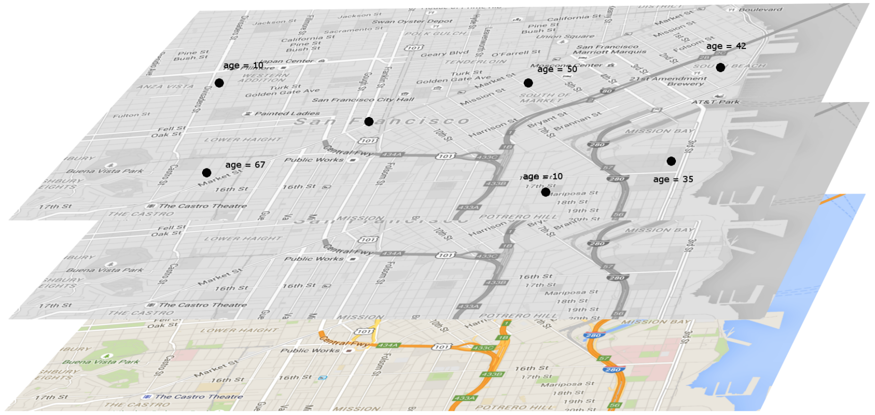
```
{
  "type": "LineString",
  "coordinates": [ [100.0, 0.0], [101.0, 1.0] ]
}
```

```
{
  "type": "Polygon",
  "coordinates": [
   [ [100.0, 0.0], [101.0, 0.0],
   [101.0, 1.0], [100.0, 1.0], [100.0, 0.0] ]
   ]
}
```

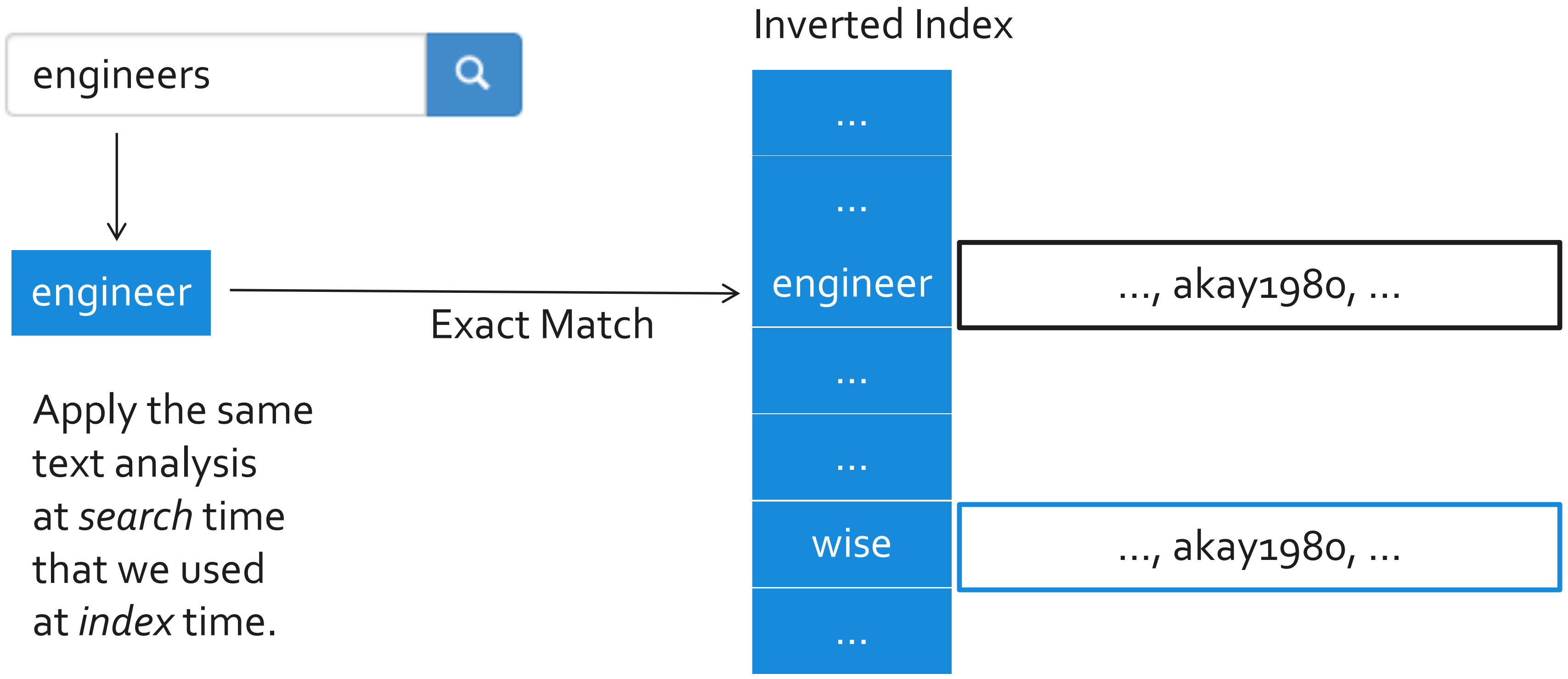


Spatial Indexes

- Combined
  - 2 dimensions for Geo-Data
  - Additional dimensions
  - e.g. all persons with an age greater than 10 in San Francisco



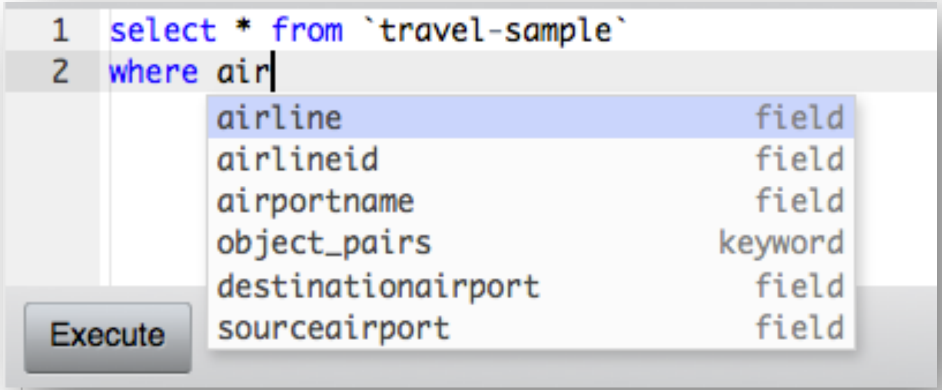
Text Indexes (Developer Preview)



Apply the same text analysis at *search* time that we used at *index* time.

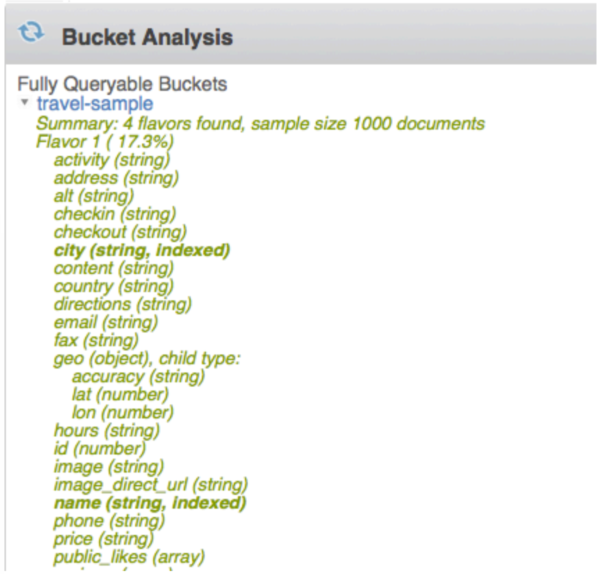
Automatic Schema Inference and Query Editor

- Automatically examines sample of documents from buckets and discovers your schema
  - Document Types and Distribution Stats
  - List of Attributes with Data Types



```
select * from `travel-sample`
where air
```

airline	field
airlineid	field
airportname	field
object_pairs	keyword
destinationairport	field
sourceairport	field



```
Fully Queryable Buckets
▾ travel-sample
  Summary: 4 flavors found, sample size 1000 documents
  Flavor 1 ( 17.3%)
    activity (string)
    address (string)
    alt (string)
    checkin (string)
    checkout (string)
    city (string, indexed)
    content (string)
    country (string)
    directions (string)
    email (string)
    fax (string)
    geo (object), child type:
      accuracy (string)
      lat (number)
      lon (number)
    hours (string)
    id (number)
    image (string)
    image_direct_url (string)
    name (string, indexed)
    phone (string)
    price (string)
    public_likes (array)
```

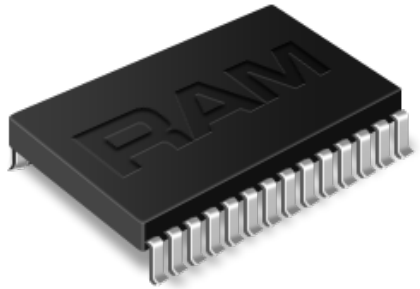
Memory Optimized Indexes

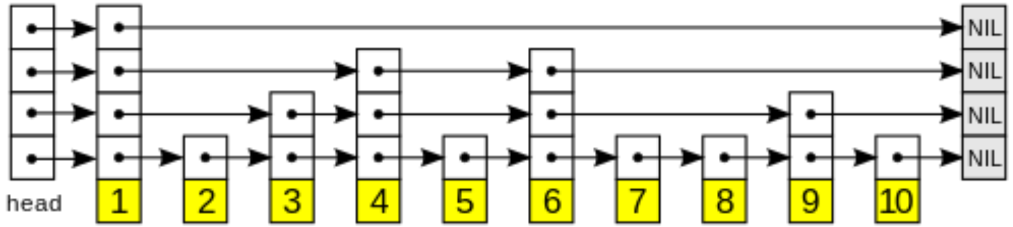
```
CREATE INDEX idx_Movies ON bucket(
     DISTINCT ARRAY r.Title FOR r IN Movies
     END
     );

SELECT Venue FROM bucket WHERE
     ANY r IN Movies SATISFIES r.Title="Fight
Club"
     END;
```

```
{
 Start_Date: "1/1/2001",
 Venue:  "AMC 55",
 Movies:
 [
   {Title : "Fight Club",
     Showtimes :
     [ { Times:[ "13:30", "14:45", "21:30"],
         3D: true},
       { Times:[ "11:30", "15:45", "20:00"],
         IMAX: true}
         …
         ]
   },
   { Title : "Sixth Sense",
     Showtimes :
     [ { Times:[ "10:30", "11:45", "13:30"],
         3D: true},
       { Times:[ "9:30", "14:45", "20:30"],
         IMAX: true}
         …
         ]
   }
 ]
}
```

Memory Optimized Indexes

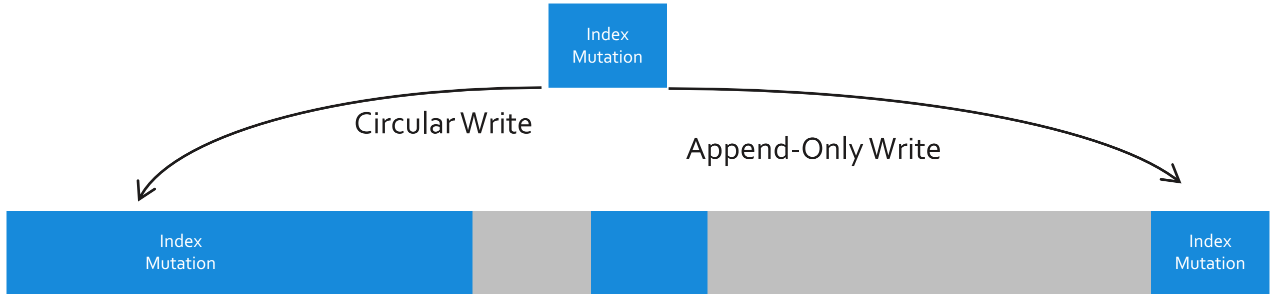
- Optimized for Memory
  - small memory footprint, optimized for lowest latency queries
- Faster Indexing
  - fresh indexes under heavy mutations with lock free index maintenance





Circular Writes

- Reduced Disk IO Requirements
  - Append-Only Writes with frequent full compaction (Version 4.1 & Earlier)
  - Circular-Reuse Writes with reduced full compactions (New in 4.5:)
  - Reused orphaned blocks in the index file
  - Reduce the need for frequent full-compactions of the index file



File for Global Secondary Indexes

High Performance Queries under strict Consistency

- t1 insert (k1, v1)
- ...
- t2 do other business logic computation
- ...
- t3 issue query on (k1,v1)

RYOW Consistency

Query execution is delayed until all indexes process mutations up to

t1

Strict Request-Time Consistency

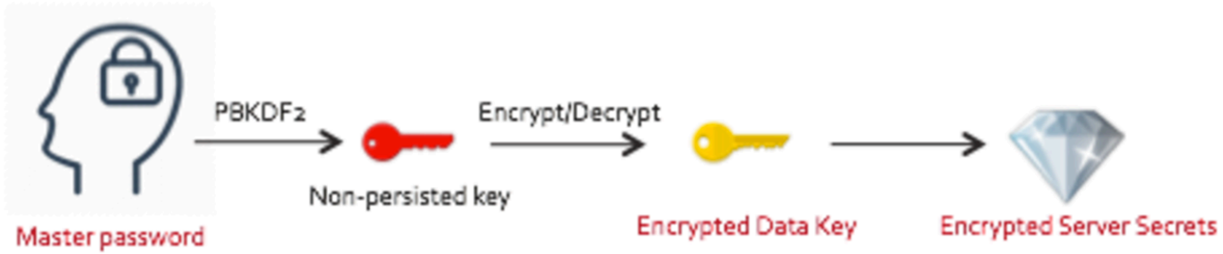
(a.k.a stale-false)

Query execution is delayed until all indexes process mutations up to

t3

4.6 Security

- P(lugable) A(uthentication) M(odule)
  - Linux only
- Secret Management
  - Own key chain for Couchbase Server secrets



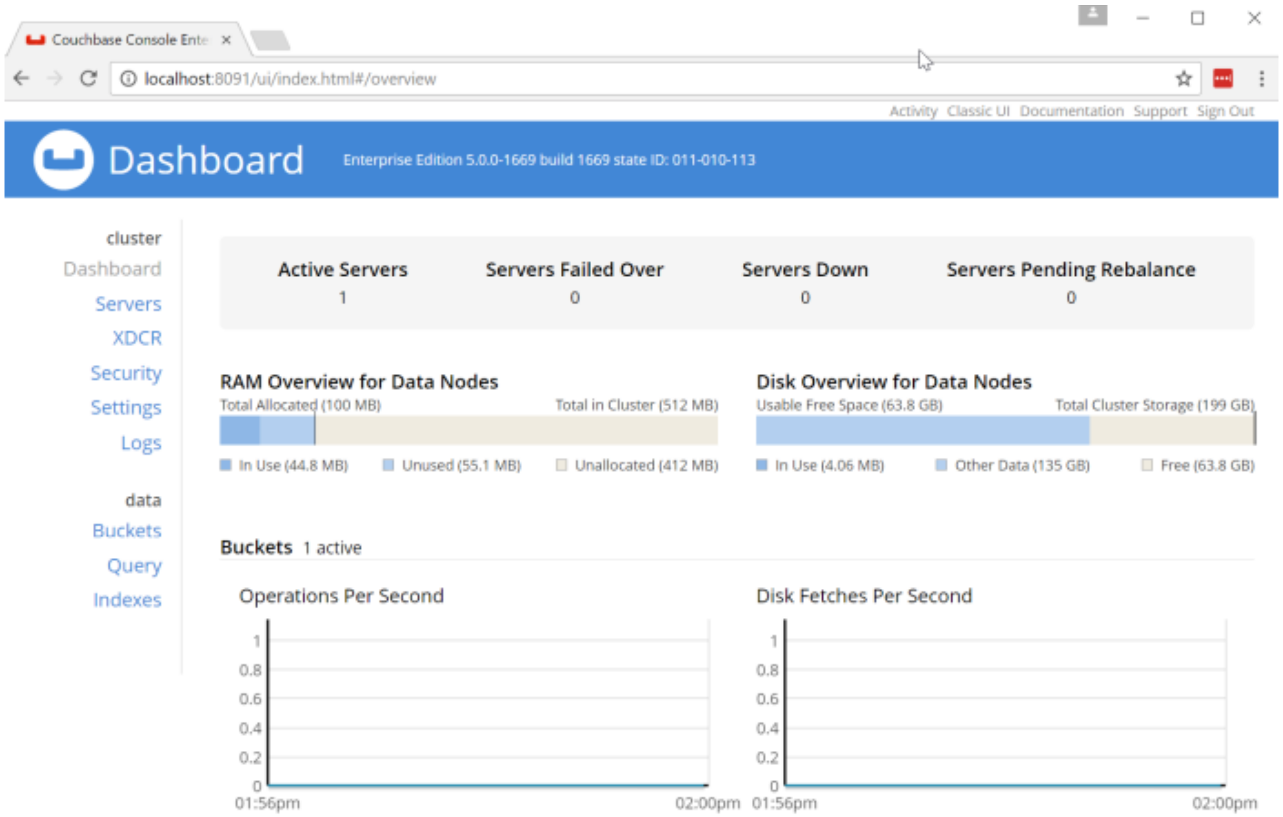
4.6 Timestamp-based Conflict Resolution for XDCR

- New conflict resolution mode
- Hybrid timestamp as the criteria to order mutations
- Higher hybrid timestamp value as the main factor to determine which document has the most recent mutation
- Hybrid timestamp is replicated across nodes within the same cluster and across clusters
- If both have the same timestamp then revision id is used

4.6 Data Structures

- **MAP**: KV structure, HashMap
- **LIST**: List of objects
- **QUEUE**: Wrapper over a list which offers FIFO semantics
- **SET**: Wrapper over a list which provides the ability to handle unique values

5.0 New UI



Enterprise Edition 5.0.0-1669 build 1669 state ID: 011-010-113

Active Servers	Servers Failed Over	Servers Down	Servers Pending Rebalance
1	0	0	0

RAM Overview for Data Nodes

Total Allocated (100 MB) Total in Cluster (512 MB)

In Use (44.8 MB) Unused (55.1 MB) Unallocated (412 MB)

Disk Overview for Data Nodes

Usable Free Space (63.8 GB) Total Cluster Storage (199 GB)

In Use (4.06 MB) Other Data (135 GB) Free (63.8 GB)

Buckets 1 active