Branch: master taste-of-riak / nodejs / Ch02-Schemas-and-Indexes / **app.js** Find file Copy path

lukebakken Ch01 for NodeJS 94826cc on Apr 3, 2015

1 contributor

275 lines (236 sloc) 7.65 KB Raw Blame History

```
'use strict';

var async = require('async');
var logger = require('winston');
var Riak = require('basho-riak-client');

// Un-comment to enable debug logging
// logger.remove(logger.transports.Console);
// logger.add(logger.transports.Console, {
//     level : 'debug',
//     colorize: true,
//     timestamp: true
// });

function checkErr(err) {
    if (err) {
        logger.error(err);
        process.exit(1);
    }
}

logger.info("Creating Data");

var customer = {
    id : '1',
    name : "John Smith",
    address : "123 Main Street",
    city : "Columbus",
    state : "Ohio",
    zip : "43210",
    phone : "+1-614-555-5555",
    createdDate : "2013-10-01 14:30:26",
};

var orders = createOrders();
var orderSummary = createOrderSummary(orders);

logger.info("Starting Client");
var node = new Riak.Node({remoteAddress: 'riak-test', remotePort: 10017});
var cluster = new Riak.Cluster({nodes: [node]});
var client = new Riak.Client(cluster);

var customersBucket = "Customers";
var ordersBucket = "Orders";
var orderSummariesBucket = "OrderSummaries";

store_objects();

function store_objects() {
    logger.info("Storing Data");
```

```javascript

    var storeFuncs = [
        function (async_cb) {
            client.storeValue({ bucket: customersBucket, key: customer.id, value: customer }, async_cb);
        },
        function (async_cb) {
            client.storeValue({
                bucket: orderSummariesBucket,
                key: orderSummary.customerId,
                value: orderSummary
            }, async_cb);
        }
    ];

    orders.forEach(function (order) {
        storeFuncs.push(
            function (async_cb) {
                client.storeValue({ bucket: ordersBucket, key: order.id, value: order }, async_cb);
            });
    });

    async.parallel(storeFuncs, function (err, rslts) {
        checkErr(err);
        fetch_objects();
    });
}

function fetch_objects() {
    logger.info("Fetching related data by shared key");

    var key = '1';
    client.fetchValue({ bucket: customersBucket, key: key }, function (err, rslt) {
        checkErr(err);
        var fetchedCustomerObj = rslt.values.shift();
        var fetchedCustomer = fetchedCustomerObj.value.toString("utf8");
        client.fetchValue({ bucket: orderSummariesBucket, key: key}, function (err, rslt) {
            checkErr(err);
            var fetchedOrderSummaryObj = rslt.values.shift();
            var fetchedOrderSummary = fetchedOrderSummaryObj.value.toString("utf8");
            logger.info("Customer     1: %s", fetchedCustomer);
            logger.info("OrderSummary 1: %s", fetchedOrderSummary);

            add_index_data();
        });
    });
}

function add_index_data() {
    logger.info("Adding Index Data");

    var funcs = [
        function (async_cb) {
            client.fetchValue({ bucket: ordersBucket, key : '1' }, function (err, rslt) {
                checkErr(err);
                var fetchedOrderObj = rslt.values.shift();
                fetchedOrderObj.addToIndex('SalespersonId_int', 9000);
                fetchedOrderObj.addToIndex('OrderDate_bin', '2013-10-01');
                client.storeValue({ value: fetchedOrderObj }, function (err, rslt) {
                    async_cb(err, rslt);
                });
            });
        },
        function (async_cb) {
            client.fetchValue({ bucket: ordersBucket, key : '2' }, function (err, rslt) {
                checkErr(err);
                var fetchedOrderObj = rslt.values.shift();
                fetchedOrderObj.addToIndex('SalespersonId_int', 9001);
                fetchedOrderObj.addToIndex('OrderDate_bin', '2013-10-15');
                client.storeValue({ value: fetchedOrderObj }, function (err, rslt) {
                    async_cb(err, rslt);
                });
            });
        },
        function (async_cb) {
            client.fetchValue({ bucket: ordersBucket, key : '3' }, function (err, rslt) {
                checkErr(err);
                var fetchedOrderObj = rslt.values.shift();
                fetchedOrderObj.addToIndex('SalespersonId_int', 9000);
                fetchedOrderObj.addToIndex('OrderDate_bin', '2013-11-03');
                client.storeValue({ value: fetchedOrderObj }, function (err, rslt) {
                    async_cb(err, rslt);
                });
            });
        }
    ];

    async.parallel(funcs, function (err, rslts) {
        checkErr(err);
        index_queries();
    });
}

function index_queries() {
    logger.info("Index Queries");

    var janesOrders = [];
    client.secondaryIndexQuery({ bucket: ordersBucket, indexName: 'SalespersonId_int', indexKey: 9000 }, function (err, rslt) {
        checkErr(err);

        if (rslt.values.length > 0) {
            Array.prototype.push.apply(janesOrders,
                    rslt.values.map(function (value) {
                        return value.objectKey;
                    })
            );
        }

        if (rslt.done) {
            logger.info("Jane's Orders: %s", janesOrders.join(', '));

            var octoberOrders = [];
            client.secondaryIndexQuery({ bucket: ordersBucket, indexName: 'OrderDate_bin',
                rangeStart: '2013-10-01', rangeEnd: '2013-10-31' }, function (err, rslt) {
                checkErr(err);

                if (rslt.values.length > 0) {
                    Array.prototype.push.apply(octoberOrders,
                            rslt.values.map(function (value) {
                                return value.objectKey;
                            })
                    );
                }

                if (rslt.done) {
                    logger.info("October's Orders: %s", octoberOrders.join(', '));

                    client_shutdown();
                }
            });
        }
    });
}

function client_shutdown() {
    client.shutdown(function (state) {
        if (state === Riak.Cluster.State.SHUTDOWN) {
            logger.info("cluster stopped");
            process.exit();
        }
    });
}

function createOrders() {
    var orders = [];

    var order1 = {
        id : '1',
        customerId : '1',
        salespersonId : 9000,
        items : [
            {
                id: "TCV37GIT4NJ",
                title: "USB 3.0 Coffee Warmer",
                price: 15.99
            },
            {
                id: "PEG10BBF2PP",
                title: "eTablet Pro, 24GB; Grey",
                price: 399.99
            },
        ],
        total : 415.98,
        orderDate : "2013-10-01 14:42:26"
    };
    orders.push(order1);

    var order2 = {
        id : '2',
        customerId : '1',
        salespersonId : 9001,
        items : [
            {
                id : "OAX19XWN0QP",
                title : "GoSlo Digital Camera",
                price : 359.99
            }
        ],
        total : 359.99,
        orderDate : "2013-10-15 16:43:16"
    };
    orders.push(order2);

    var order3 = {
        id : '3',
        customerId : '1',
        salespersonId : 9000,
        items : [
            {
                id : "WYK12EPU5EZ",
                title : "Call of Battle : Goats - Gamesphere 4",
                price : 69.99
            },
            {
                id : "TJB84HAA8OA",
                title : "Bricko Building Blocks",
                price : 4.99
            }
        ],
        total : 74.98,
        orderDate : "2013-11-03 17:45:28"
    };
    orders.push(order3);

    return orders;
}

function createOrderSummary(orders) {

    var orderSummary = {
        customerId : '1',
        summaries : []
    };

    orders.forEach(function (order) {
        orderSummary.summaries.push({
            orderId : order.id,
            total : order.total,
            orderDate : order.orderDate
        });
    });

    return orderSummary;
}

```