

# COUCHDB

## INFORMACION

### FORMATO

Presencial

En Sitio

A partir de 3 participantes

### DURACIÓN:

20 Horas

5 días

Lunes a Viernes

### DIRIGIDO A

Analistas y programadores  
que requieren procesar y  
analizar grandes volúmenes  
de datos

### REQUISITOS:

Conocimientos básicos de  
Linux.

### MATERIAL:

Manual Oficial

DVD de la distribución

Linux más reciente

### DOCUMENTO

Diploma expedido por

PLCT S.A. DE C.V.

## DESCRIPCION GENERAL

Curso orientado a usuarios y desarrolladores de aplicaciones que requieren acceder a información almacenada en un modelo de documentos NoSQL.

## OBJETIVOS

Que el estudiante aprenda y aplique las técnicas y metodologías de procesamiento de información almacenada en un modelo de documentos NoSQL.



## Part I. Introduction

**1. Why CouchDB?**

- Relax
- A Different Way to Model Your Data
- A Better Fit for Common Applications
  - Self-Contained Data
  - Syntax and Semantics
- Building Blocks for Larger Systems
  - CouchDB Replication
- Local Data Is King
- Wrapping Up

**2. Eventual Consistency**

- Working with the Grain
- The CAP Theorem
- Local Consistency
  - The Key to Your Data
  - No Locking
  - Validation
- Distributed Consistency
  - Incremental Replication
  - Case Study
- Wrapping Up

**3. Getting Started**

- All Systems Are Go!
- Welcome to Futon
- Your First Database and Document
- Running a Query Using MapReduce
- Triggering Replication
- Wrapping Up

**4. The Core API**

- Server
- Databases
- Documents
  - Revisions
  - Documents in Detail
- Replication
- Wrapping Up

## Part II. Developing with CouchDB

**5. Design Documents**

- Document Modeling
- The Query Server
- Applications Are Documents
- A Basic Design Document
- Looking to the Future

**6. Finding Your Data with Views**

- What Is a View?
- Efficient Lookups
  - Find One
  - Find Many
  - Reversed Results
- The View to Get Comments for Posts
- Reduce/Rereduce
  - Lessons Learned
- Wrapping Up

**7. Validation Functions**

- Document Validation Functions
- Validation's Context
- Writing One
  - Type
  - Required Fields
  - Timestamps
  - Authorship
- Wrapping Up

**8. Show Functions**

- The Show Function API
- Side Effect-Free
- Design Documents
- Querying Show Functions
  - Design Document Resources
  - Query Parameters
  - Accept Headers
- Etags
- Functions and Templates
  - The !json Macro
  - The !code Macro
- Learning Shows
- Using Templates
- Writing Templates

**9. Transforming Views with List Functions**

- Arguments to the List Function
- An Example List Function
- List Theory
- Querying Lists
- Lists, Etags, and Caching

## Part III. Example Application

**10. Standalone Applications**

- Use the Correct Version
- Portable JavaScript
- Applications Are Documents
- Standalone
- In the Wild
- Wrapping Up

**11. Managing Design Documents**

- Working with the Example Application
- Installing CouchApp
- Using CouchApp
- Download the Sofa Source Code
  - CouchApp Clone
  - ZIP and TAR Files
  - Join the Sofa Development Community on GitHub
  - The Sofa Source Tree
- Deploying Sofa
  - Pushing Sofa to Your CouchDB
  - Visit the Application
- Set Up Your Admin Account
  - Deploying to a Secure CouchDB
- Configuring CouchApp with .couchapprc

**12. Storing Documents**

- JSON Document Format
- Beyond \_id and \_rev: Your Document Data
- The Edit Page
  - The HTML Scaffold
- Saving a Document
  - Validation
  - Save Your First Post
- Wrapping Up

**13. Showing Documents in Custom Formats**

- Rendering Documents with Show Functions
  - The Post Page Template
- Dynamic Dates

**14. Viewing Lists of Blog Posts**

- Map of Recent Blog Posts
- Rendering the View as HTML Using a List Function
  - Sofa's List Function
  - The Final Result

**Part IV. Deploying CouchDB****15. Scaling Basics**

- Scaling Read Requests
- Scaling Write Requests
- Scaling Data
- Basics First

**16. Replication**

- The Magic
- Simple Replication with the Admin Interface
- Replication in Detail
- Continuous Replication
- That's It?

**17. Conflict Management**

- The Split Brain
- Conflict Resolution by Example
- Working with Conflicts

- Deterministic Revision IDs
- Wrapping Up

**18. Load Balancing**

- Having a Backup

**19. Clustering**

- Introducing CouchDB Lounge
- Consistent Hashing
  - Redundant Storage
  - Redundant Proxies
- View Merging
- Growing the Cluster
  - Moving Partitions
  - Splitting Partitions

**Part V. Reference****20. Change Notifications**

- Polling for Changes
- Long Polling
- Continuous Changes
- Filters
- Wrapping Up

**21. View Cookbook for SQL Jockeys**

- Using Views
  - Defining a View
  - Querying a View
  - MapReduce Functions
- Look Up by Key
- Look Up by Prefix
- Aggregate Functions
- Get Unique Values
- Enforcing Uniqueness

**22. Security**

- The Admin Party
  - Creating New Admin Users
  - Hashing Passwords
- Basic Authentication
  - Update Validations Again
- Cookie Authentication
- Network Server Security

**23. High Performance**

- Good Benchmarks Are Non-Trivial
- High Performance CouchDB
  - Hardware
  - An Implementation Note
- Bulk Inserts and Mostly Monotonic DocIDs
  - Optimized Examples: Views and Replication
- Bulk Document Inserts
- Batch Mode
- Single Document Inserts
- Hovercraft
- Trade-Offs
  - But...My Boss Wants Numbers!
  - A Call to Arms

**24. Recipes**

Banking

Accountants Don't Use Erasers

Wrapping Up

Ordering Lists

A List of Integers

A List of Floats

Pagination

Example Data

A View

Setup

Slow Paging (Do Not Use)

Fast Paging (Do Use)

Jump to Page