📖Capítulo 5 – CouchDB

Introdução – Relaxando no Sofá (*Relaxing on the Couch*)

O **CouchDB**, desenvolvido pela Apache Foundation, é um banco de dados **NoSQL orientado a documentos**. Diferente do MongoDB, que prioriza flexibilidade e escalabilidade em clusters, o CouchDB nasceu para resolver o problema de **replicação confiável, sincronização offline e tolerância a falhas**.

O lema do projeto é: **"Relax."** – refletindo a ideia de simplicidade para o desenvolvedor em ambientes distribuídos.

Principais características

- Armazenamento em documentos JSON (sem esquema fixo).
- API RESTful nativa (operações via HTTP: GET, POST, PUT, DELETE).
- MVCC (Multiversion Concurrency Control) → documentos possuem `_id` (identificador único) e `_rev` (versão do documento).
- Replicação e sincronização nativas.
- Alta tolerância a falhas (ideal para ambientes offline-first).

Estrutura de documento

```
{
  "_id": "cliente:1001",
  "_rev": "1-9c65296036141e575d32ba9c034dd3ee",
  "nome": "Douglas",
  "idade": 30,
  "email": "douglas@example.com",
  "enderecos": [
    { "tipo": "residencial", "cidade": "São Paulo" },
    { "tipo": "trabalho", "cidade": "Campinas" }
  ]
}
```

Dia 1 – CRUD, Fauxton e cURL Redux

O CouchDB expõe uma **API HTTP/RESTful**. Qualquer cliente HTTP (curl, Postman, navegador) pode interagir com ele.

Fauxton (interface web)

- Acessível via `http://127.0.0.1:5984/_utils`.
- Permite criar bancos, inserir documentos, consultar e configurar replicações.

Criando banco

```
curl -X PUT http://127.0.0.1:5984/biblioteca
```

Criando documentos

```
# POST (id gerado automaticamente)
curl -X POST http://127.0.0.1:5984/biblioteca
  -H "Content-Type: application/json"
  -d '{"titulo": "1984", "autor": "George Orwell", "ano": 1949}'

# PUT (id definido manualmente)
curl -X PUT http://127.0.0.1:5984/biblioteca/livro_1984
  -H "Content-Type: application/json"
  -d '{"titulo": "1984", "autor": "George Orwell", "ano": 1949}'
```

Lendo documentos

```
curl -X GET http://127.0.0.1:5984/biblioteca/livro_1984
```

Atualizando documentos

```
curl -X PUT http://127.0.0.1:5984/biblioteca/livro_1984
  -H "Content-Type: application/json"
  -d '{"_id": "livro_1984", "_rev": "1-967a00dff5e02add41819138abb3284d",
"titulo": "1984", "autor": "George Orwell", "ano": 1950}'
```

Excluindo documentos

```
curl -X DELETE http://127.0.0.1:5984/biblioteca/livro_1984?rev=2-
c65ddf34e8c82fe9d6f91f5a09c1f9c2
```

Dia 2 – Criando e Consultando Views

As **Views** substituem o SQL no CouchDB. Elas são baseadas em **MapReduce com JavaScript** e ficam armazenadas em **design documents**.

Exemplo de view (livros por autor)

```
{
  "_id": "_design/livros",
  "views": {
    "por_autor": {
```

```
      "map": "function(doc) { if(doc.autor) { emit(doc.autor,
doc.titulo); } }"
    }
  }
}
```

Consulta

```
curl -X GET http://127.0.0.1:5984/biblioteca/_design/livros/_view/por_autor
```

View com reduce (quantidade por autor)

```
{
  "_id": "_design/livros",
  "views": {
    "quantidade_por_autor": {
      "map": "function(doc) { if(doc.autor) { emit(doc.autor, 1); } }",
      "reduce": "function(keys, values, rereduce) { return sum(values); }"
    }
  }
}
```

Consulta com agrupamento:

```
curl -X GET "http://127.0.0.1:5984/biblioteca/_design/livros/_view/
quantidade_por_autor?group=true"
```

Parâmetros úteis

- `?key="George Orwell"` → resultados para um autor.
- `?startkey="A"&endkey="M"` → intervalo.
- `?limit=10` → limitar resultados.
- `?descending=true` → ordem inversa.

Dia 3 – Advanced Views, Changes API e Replicating Data

Advanced Views

Suporte a intervalos, paginação e ordenação com parâmetros (`startkey`, `endkey`, `limit`, `descending`).

Changes API

Feed em tempo real de alterações:

```
curl -X GET http://127.0.0.1:5984/biblioteca/_changes
```

Replicação

Coração do CouchDB: replicação local/remota, uni ou bidirecional.

```
curl -X POST http://127.0.0.1:5984/_replicate
  -H "Content-Type: application/json"
  -d '{"source": "biblioteca", "target": "http://servidor_remoto:5984/
biblioteca", "create_target": true}'
```

Conflitos

- CouchDB **não bloqueia** conflitos.
- Mantém múltiplas versões (`_rev`).
- Uma versão é eleita como principal, as demais ficam disponíveis para resolução manual.

Casos de uso

- Aplicativos móveis offline-first.
- Filiais distribuídas.
- Redes instáveis.

Wrap-Up do Capítulo 5 – CouchDB

- Também orientado a documentos JSON, mas com foco em **replicação e sincronização**.
- Usa **HTTP/REST** como interface universal.
- Versionamento com **MVCC**.
- Views baseadas em **MapReduce** para consultas indexadas.
- Replicação e resolução de conflitos como parte essencial do modelo.

👉Ideal para cenários onde a **conectividade é limitada** e a **consistência eventual** é aceitável.