Técnicas para Big Data

Clase 04: JSON y NoSQL

Hasta ahora

- Bases de datos relacionales
- SQL

NoSQL

Término común para denominar bases de datos con:

- Menos restricciones que el modelo relacional
- Menos esquema
- Más distribución

BD Orientadas a Documentos

Especializadas en documentos

- CouchDB, MongoDB (estas y otras BD almacenan sus datos en documentos JSON)
- JSON no es el único estándar de documentos (por ejemplo, existe también XML)

BD Key - Value

- Son grandes tablas de hash persistentes
- Esta categoría es difusa, pues muchas de las aplicaciones de otros tipos de BD usan key - value y hashing hasta cierto punto

BD de Grafos y RDF

Especializadas para guardar relaciones

- En general, almacenan sus datos como property graphs
- Algunos ejemplos son Neo4J, Virtuoso, Jena, Blazegraph

JSON

Su nombre viene de JavaScript Object Notation

Estándar de intercambio de datos semiestructurados / datos en la Web

- JSON se acopla muy bien a los lenguajes de programación

JSON

Ejemplo


```
{
  "statuses": [
    {
      "id": 725459373623906304,
      "text": "@visitlondon: Have you been to any of these
               quirky London museums? https://t.co/tnrar8UttZ",
      "retweeted_status": {
        "metadata": {
          "result_type": "recent",
          "iso_language_code": "en"
        },
        "retweet_count": 239,
        "retweeted": false
      }
    }
  ]
}
```

JSON

La base son los pares key - value

```
{
  “nombre": “Matías”
}
```

Valores pueden ser:

- Números
- Strings (entre comillas)
- Valores booleanos
- Arreglos (por definir)
- Objetos (por definir)
- `null`

JSON

Sintaxis

Los objetos se escriben entre **{}** y contienen una cantidad arbitraria de pares key - value

```
{
  “nombre": “Matías”, “apellido": “Jünemann”
}
```

JSON

Sintaxis

Los arreglos se escriben entre [ ] y contienen valores

```
{
  “profesores”: [
    {“nombre”: “Juan”, “apellido": “Reutter”},
    {“nombre”: “Cristian”, “apellido": “Riveros”},
    {“nombre”: “Marcelo”, “apellido": “Arenas”}
  ]
}
```

JSON vs SQL

SQL:

- Esquema de datos
- Lenguajes de consulta independientes del código

JSON:

- Más flexible, no hay que respetar necesariamente un esquema
- Más tipos de datos (como arreglos)
- Human - Readable

JSON

Lenguaje de consultas

Hay intentos de lenguajes de consulta para objetos JSON que usen su estructura de árbol:

- Por ejemplo, JSONPath

Importante: JSON está ahí para los programadores que NO buscan separar datos del código

JSON

Lenguaje de consultas

¿Por qué necesitamos esquemas para JSON?

- JSON Schema: propuesta toma fuerza el 2013 - 2014
- Harta investigación en el DataLab UC

JSON Schema

```
{
  “first_name”: “Alexis”,
  “last_name”: “Sánchez”,
  “age”: 28,
  “club”: {
    “name”: “Arsenal FC”,
    “founded”: 1886,
  }
  “first_club”: “Cobreloa”,
  “va_al_mundial”: false
}
```

JSON Schema

```
{
    "type": "object",
    "properties": {
        "first_name": { "type": "string" },
        "last_name": { "type": "string" },
        "age": { "type": "integer" },
        "club": {
            "type": "object",
            "properties": {
                "name": { "type": "string" },
                "founded": { "type": "integer" }
            },
            "required": ["name"]
        }
        “first_club": { "type": "string" },
        “va_al_mundial”: { "type": “boolean" },
    },
    "required": ["first_name", "last_name", "age", "club"]
}
```

BD Key - Value

Independientemente del esquema

- Arquitectura almacena información por medio de pares
- Cada par tiene una llave (identificador) y un valor

BD de documentos

Especializadas en documentos: almacenan muchos documentos JSON

- Si quiero libros: un documento JSON por libro
- Si quiero personas: un documento JSON por persona

Notar que esto es altamente jerárquico

BD de documentos

Qué hacen bien:

- Si quiero un libro o persona en particular
- Cruce de información **simple**

Muy útiles a la hora de desplegar información en la web

BD de documentos

Pueden verse como un cache de una BD relacional
¿Por qué?

BD de documentos

Si quiero cruzar información:

- Documentos de alumnos
- Documentos de ramos

Muy fácil:

- Todos los alumnos que toman un ramo ¿Cómo lo hago?
- Los ramos con más alumnos ¿Cómo lo hago?

BD de documentos

Si quiero cruzar información:

- Documentos de alumnos
- Documentos de ramos

No tan fácil:

- Los ramos con más alumnos en ingeniería ¿Cómo lo hago?
- Si el join es complejo, la estructura jerárquica es un impedimento

Teorema CAP

Plantea que para una base de datos distribuida es imposible mantener simultáneamente las tres características:

- Consistency
- Availability
- Partition Tolerance

BD Documentos vs Teorema CAP

- Distintas aplicaciones en una misma base de datos acceden a distintos documentos al mismo tiempo
- En general diseñadas para montar varias instancias que (en teoría) tienen la misma información
- Propagan updates en forma descoordinada

Proveen "**Consistencia Eventual**"

Consistencia Eventual

La consistencia eventual puede generar problemas

Si dos aplicaciones intentan acceder al mismo documento en MongoDB, estas pueden ser versiones diferentes del documento

Técnicas para Big Data

Clase 04: JSON y NoSQL