

NoSQL II: base de datos en grafos

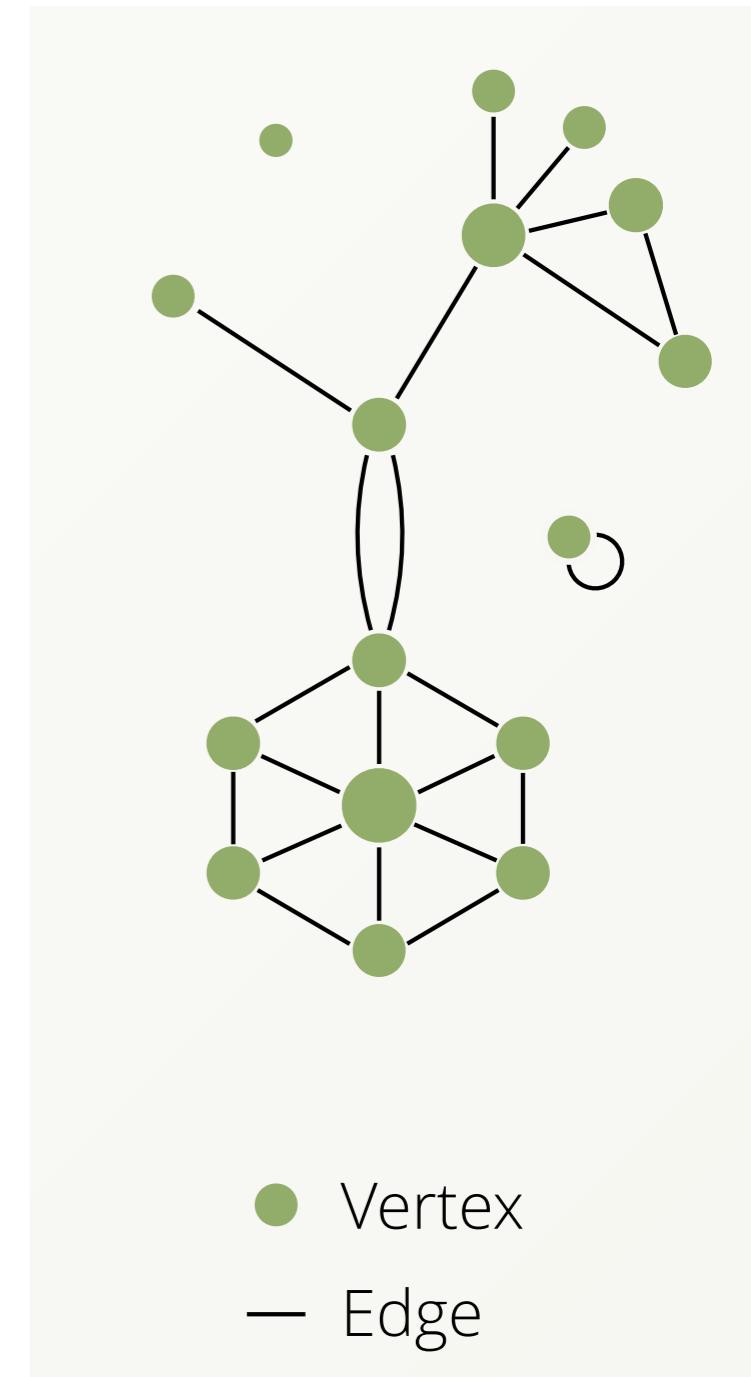
Alex Di Genova

07/11/2022

Grafos

Que es un grafo?

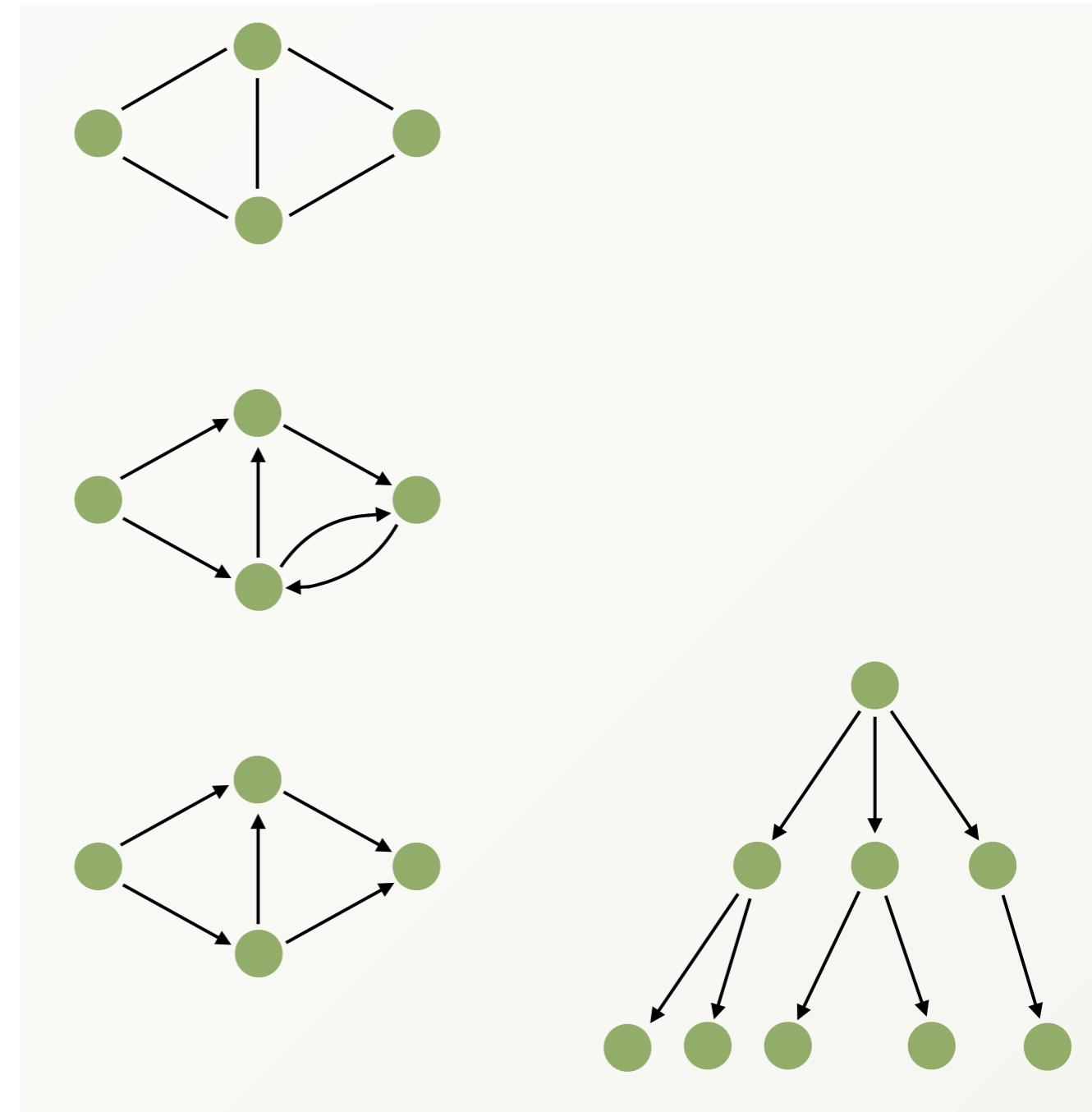
- Existen múltiples definiciones y tipos. Una breve reseña:
- En matemáticas discretas, un grafo se define como un **conjunto de vértices y aristas**. En computación se considera un **tipo de datos abstracto** que es muy útil para representar conexiones o relaciones.
 - A diferencia de las estructuras de datos tabulares de los sistemas de bases de datos relacionales, que irónicamente son muy limitados para expresar relaciones.
- Una buena metáfora de los grafos es pensar en los nodos como círculos y las aristas como líneas o arcos. Los términos nodo y vértice se usan indistintamente.
 - Por lo general, los vértices están conectados por aristas, formando un grafo.
 - Los vértices no tienen que estar conectados, pero también pueden estar conectados con más de un vértice a través de múltiples aristas.
 - También puede encontrar vértices conectados entre sí.



Grafos

Tipos de Grafos

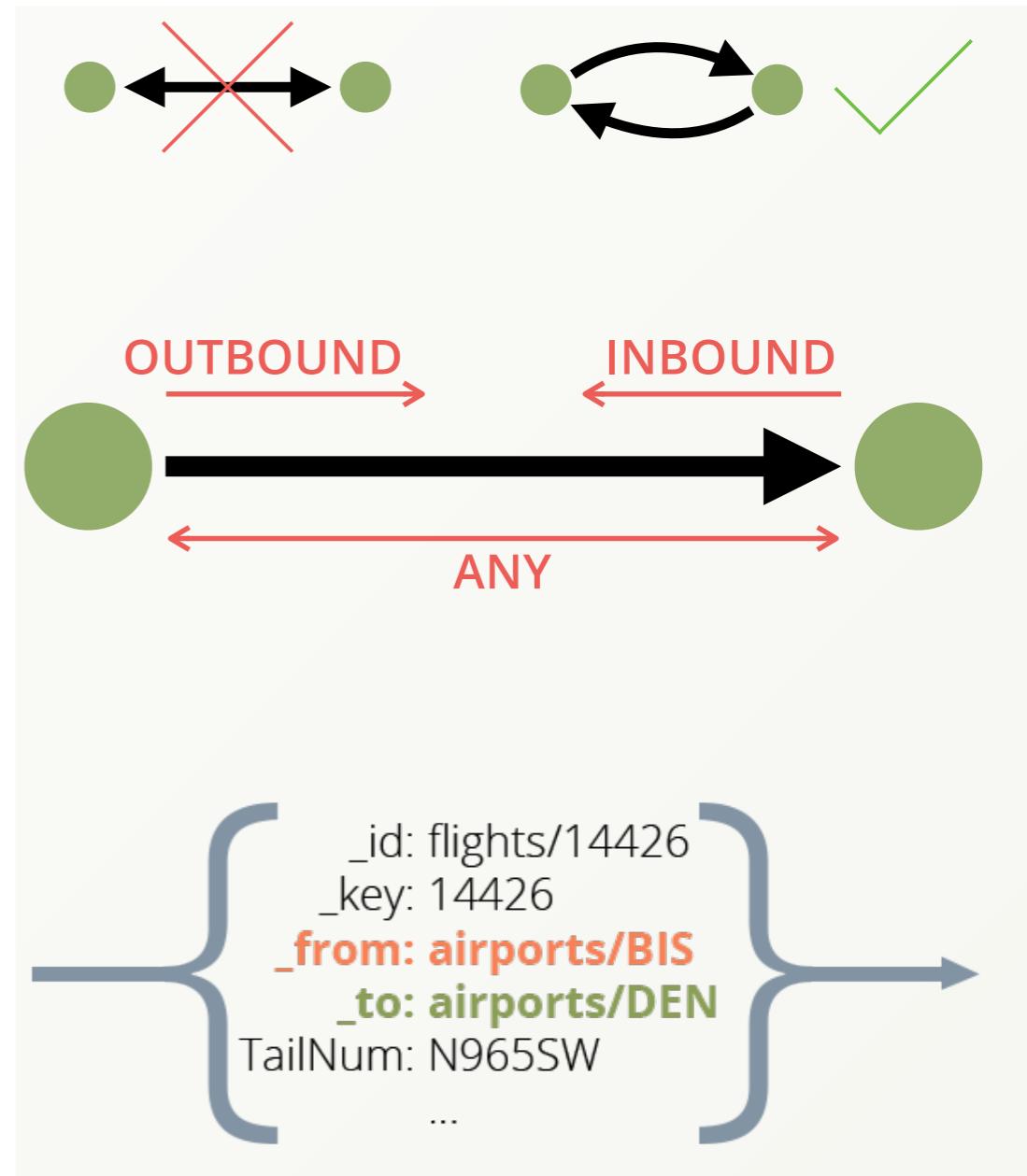
- **No dirigido:** las aristas conectan pares de nodos sin tener una noción de dirección
- **Dirigido:** las aristas (arco) tienen una dirección asociada.
- DAG – Grafo acíclico dirigido: las aristas tienen una dirección y no hay ciclos.
 - En el caso más simple, esto significa que si tenemos los vértices A y B y una arista de A a B, entonces no puede haber otra arista de B a A.
 - Un ejemplo para un DAG es un árbol.



DB en Grafos

ArangoDB

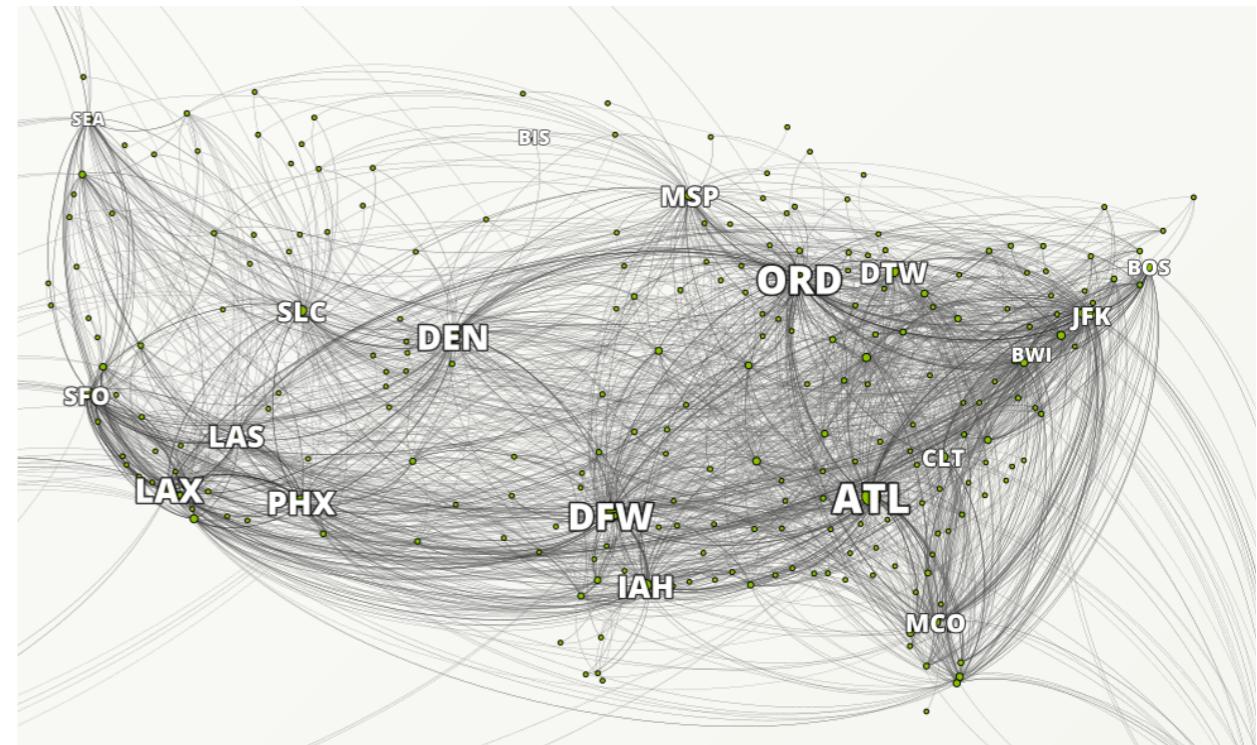
- En ArangoDB, cada arista tiene una sola dirección, no puede apuntar en ambos sentidos a la vez. Este modelo también se conoce como **grafo dirigido**.
- Pero la dirección se puede ignorar (seguir en CUALQUIER dirección) cuando movemos en el grafo, o seguir las aristas en dirección inversa (INBOUND) en lugar de ir en la dirección a la que realmente apuntan (OUTBOUND). Moverse en un grafo se llama recorrido.
- ArangoDB permite almacenar todo tipo de grafos en diferentes formas y tamaños, con y sin ciclos. Podemos guardar uno o más aristas entre dos vértices o incluso con el mismo vértice.
- Las aristas son documentos JSON completos, por lo tanto podemos almacenar tanta información como deseemos/necesitemos.



DB en grafos

Aeropuertos y vuelos

- Aeropuertos : 3,375
- Vuelos : 286,463
- Consultas:
 - Listar todos los vuelos que salen de **JFK** (aeropuerto de Nueva York)
 - Listar todos los vuelos que aterrizan en LAX (aeropuerto de Los Ángeles) el 5 de enero.
 - ¿Cuál es la cantidad mínima de escalas para volar desde BIS (Aeropuerto Municipal de Bismarck en Dakota del Norte) a LAX?

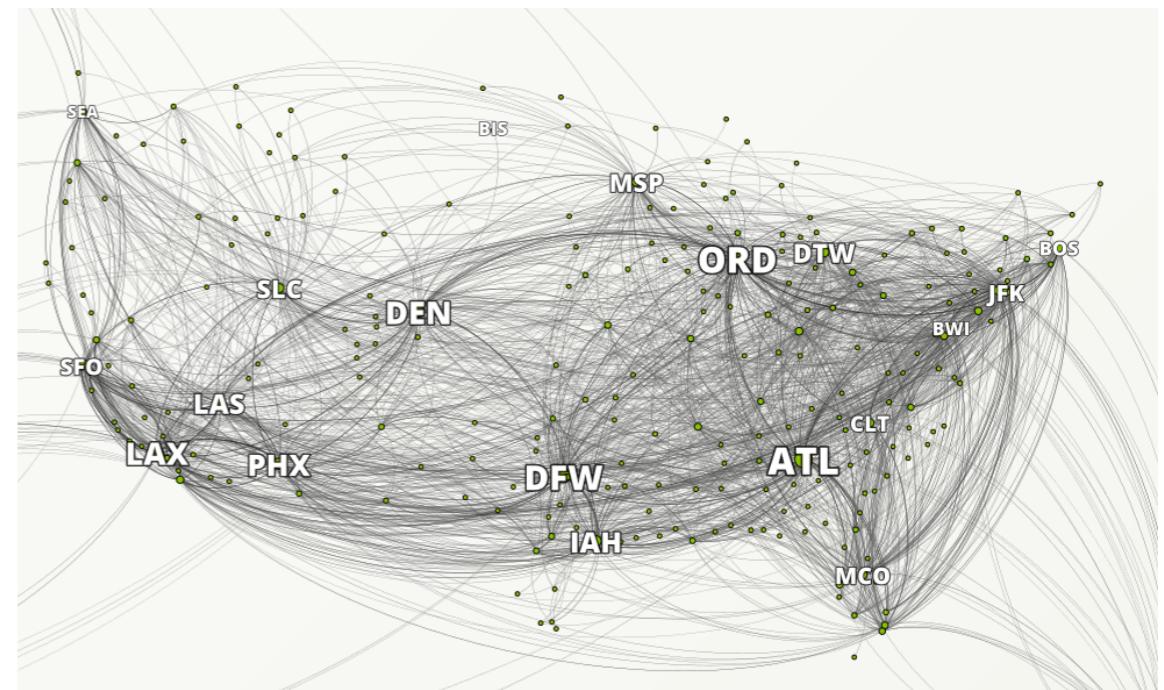


Vuelos&Aeropuertos DB

Documentos aeropuertos

- Atributos

- _key : código abreviación
- _id : nombre colección+"/"+_key
- Lat y long : latitud y longitud
- Vip : ¿Aeropuerto con salón premium?



Aeropuertos

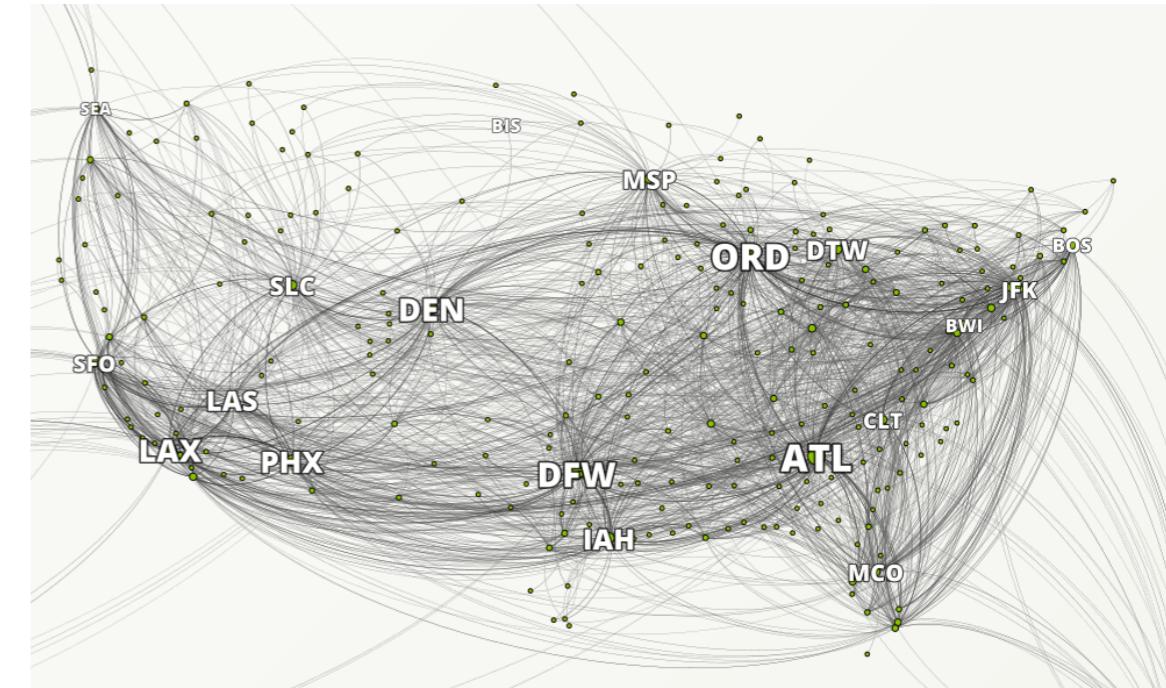
key	name	city	state	country	lat	long	vip
00M	Thigpen	Bay Springs	MS	USA	31.95376472	-89.23450472	FALSE
00R	Livingston Municipal	Livingston	TX	USA	30.68586111	-95.01792778	FALSE
00V	Meadow Lake	Colorado Springs	CO	USA	38.94574889	-104.5698933	FALSE
01G	Perry-Warsaw	Perry	NY	USA	42.74134667	-78.05208056	FALSE
01J	Hilliard Airpark	Hilliard	FL	USA	30.6880125	-81.90594389	FALSE
01M	Tishomingo County	Belmont	MS	USA	34.49166667	-88.20111111	FALSE
02A	Gragg-Wade	Clanton	AL	USA	32.85048667	-86.61145333	FALSE
02C	Capitol	Brookfield	WI	USA	43.08751	-88.17786917	FALSE
02G	Columbiana County	East Liverpool	OH	USA	40.67331278	-80.64140639	FALSE

Vuelos&Aeropuertos DB

Documentos vuelos

- Atributos

- `_from` : aeropuerto de origen (id_aeropuerto)
- `_to` : aeropuerto destino (id_aeropuerto)
- Year, Month, Day, DayofWeek.
- DepTime : Hora de salida
- ArrTime : Hora de llegada
- FlightNum: Numero de vuelo
- TailNum : Numero de Avión.
- UniqueCarrier : Código de operador.
- Distance : Distancia de vuelo en millas.



Vuelos

<code>_from</code>	<code>_to</code>	Year	Month	Day	DayOfWeek	DepTime	ArrTime	UniqueCarrier	FlightNum	TailNum	Distance
airports/ATL	airports/CHS	2008	1	1	2	2	57	FL	579	N937AT	259
airports/CLE	airports/SAT	2008	1	1	2	3	230	XE	2895	N14158	1241
airports/IAD	airports/CLE	2008	1	1	2	5	132	YV	7185	N592ML	288
airports/JFK	airports/PBI	2008	1	1	2	8	332	B6	859	N505JB	1028
airports/CVG	airports/MHT	2008	1	1	2	9	215	OH	5169	N669CA	741
airports/JFK	airports/SFO	2008	1	1	2	11	327	UA	9	N555UA	2586
airports/MIA	airports/TPA	2008	1	1	2	14	105	AA	1831	N3CHAA	204
airports/CVG	airports/GSO	2008	1	1	2	25	148	OH	5448	N398CA	330
airports/FLL	airports/JFK	2008	1	1	2	26	250	B6	878	N656JB	1069

Vuelos&Aeropuertos DB

Ejemplos de Documentos en JSON

Aeropuertos

```
{  
  "_key": "JFK",  
  "_id": "airports/JFK",  
  "_rev": "_Y0008KG-_T",  
  "name": "John F Kennedy Intl",  
  "city": "New York",  
  "state": "NY",  
  "country": "USA",  
  "lat": 40.63975111,  
  "long": -73.77892556,  
  "vip": true  
}
```

```
{  
  "_key": "BIS",  
  "_id": "airports/BIS",  
  "_rev": "_Y0SrLBe--r",  
  "name": "Bismarck Municipal",  
  "city": "Bismarck",  
  "state": "ND",  
  "country": "USA",  
  "lat": 46.77411111,  
  "long": -100.7467222,  
  "vip": false  
}
```

Vuelos

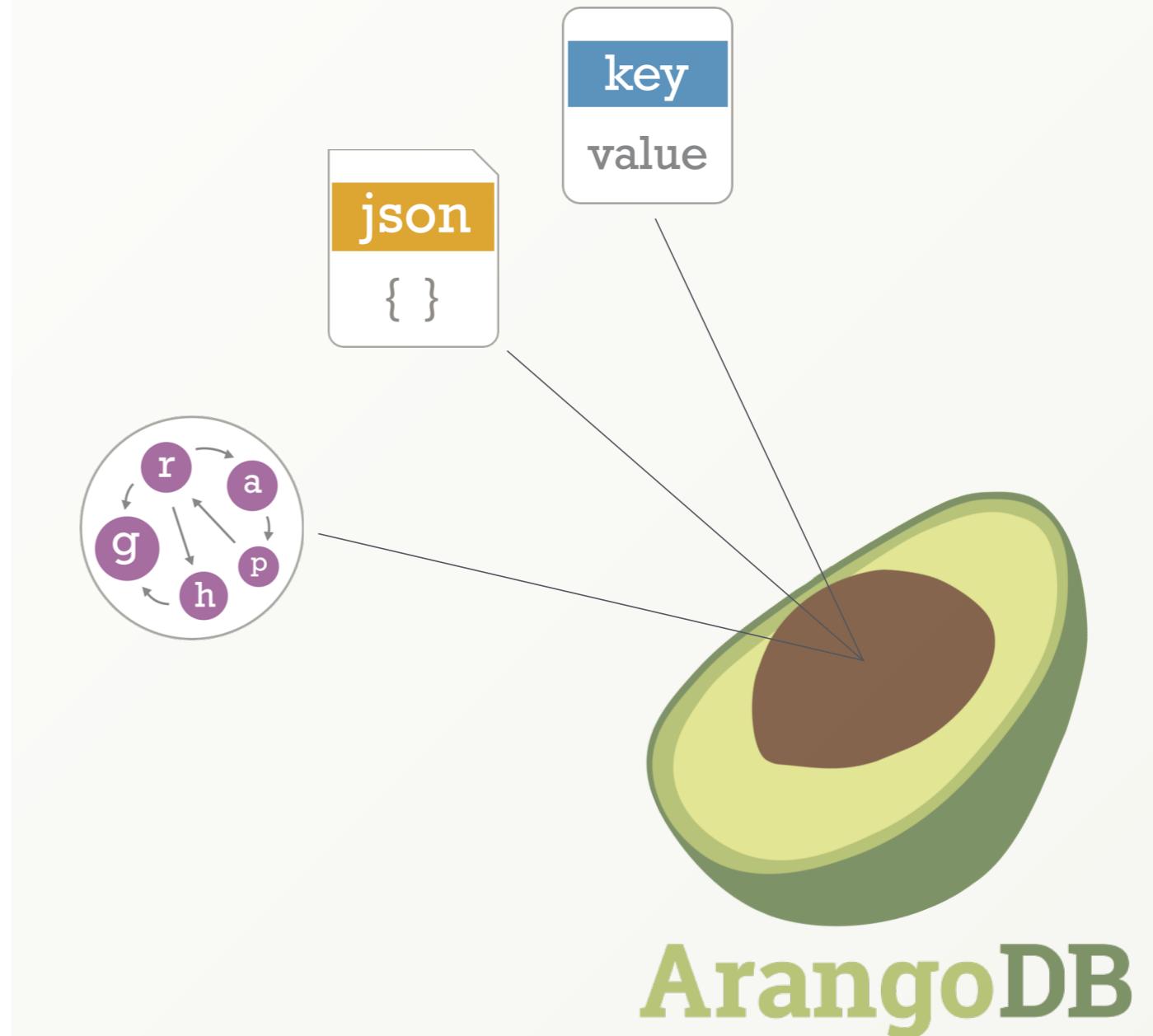
```
{  
  "_key": "25471",  
  "_id": "flights/25471",  
  "_from": "airports/BIS",  
  "_to": "airports/MSP",  
  "_rev": "_Y008JXG--f",  
  "Year": 2008,  
  "Month": 1,  
  "Day": 2,  
  "DayOfWeek": 3,  
  "DepTime": 1055,  
  "ArrTime": 1224,  
  "DepTimeUTC": "2008-01-02T16:55:00.000Z",  
  "ArrTimeUTC": "2008-01-02T18:24:00.000Z",  
  "UniqueCarrier": "9E",  
  "FlightNum": 5660,  
  "TailNum": "85069E",  
  "Distance": 386  
}
```

```
{  
  "_key": "71374",  
  "_id": "flights/71374",  
  "_from": "airports/JFK",  
  "_to": "airports/DCA",  
  "_rev": "_Y008LYG--N",  
  "Year": 2008,  
  "Month": 1,  
  "Day": 4,  
  "DayOfWeek": 5,  
  "DepTime": 1604,  
  "ArrTime": 1724,  
  "DepTimeUTC": "2008-01-04T21:04:00.000Z",  
  "ArrTimeUTC": "2008-01-04T22:24:00.000Z",  
  "UniqueCarrier": "MQ",  
  "FlightNum": 4755,  
  "TailNum": "N854AE",  
  "Distance": 213  
}
```

ArangoDB

Motor muti-modelo

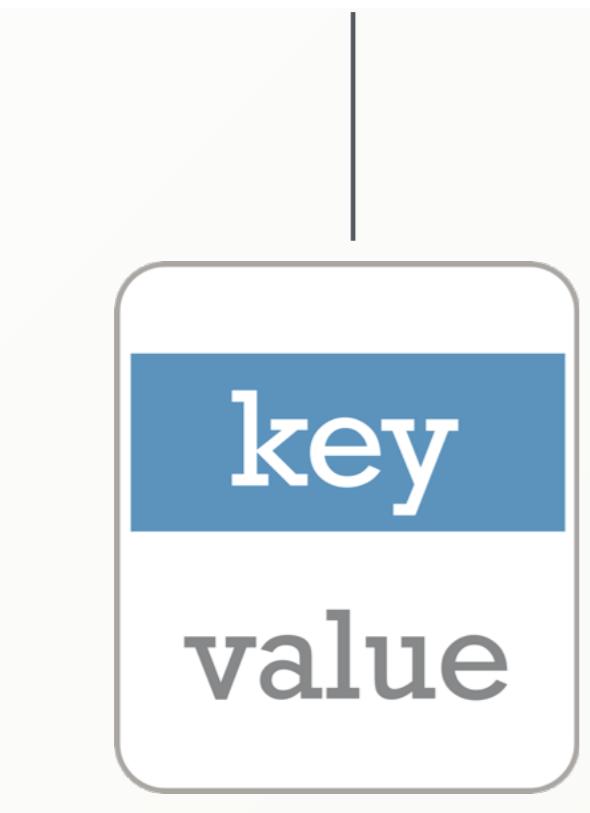
- Características únicas de AQL:
 - Posibilidad de combinar los 3 modelos de datos en una sola consulta.
 - Podemos combinar uniones (joins), recorridos (traversal), filtros, operaciones geoespaciales y agregaciones en nuestras consultas



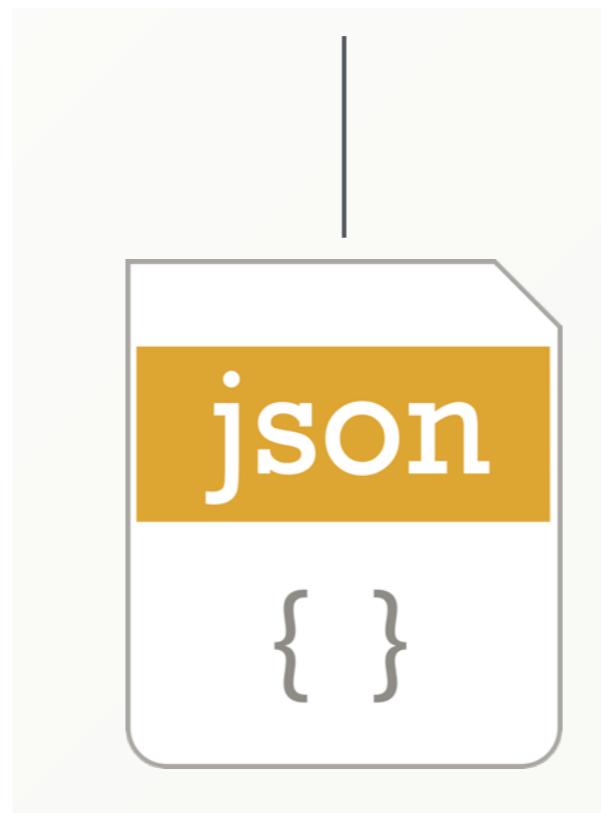
Implementacion Multi-Modelo

¿Cómo es implementado el multi-modelo en Arango?

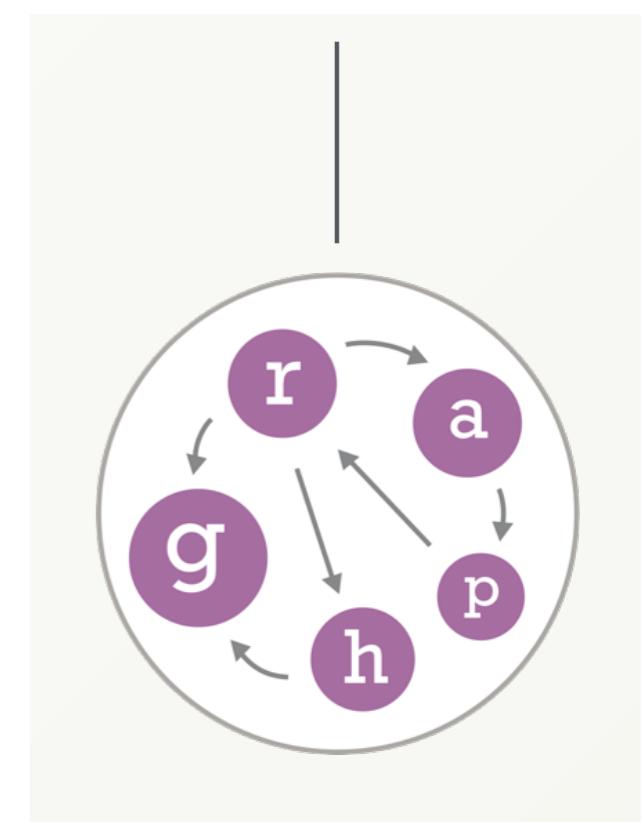
Si almacenamos un documento JSON y lo tratamos como un valor de una clave primaria, entonces tenemos documento con clave/valor.



ArangoDB es un motor de datos orientado a documentos que utiliza claves primarias

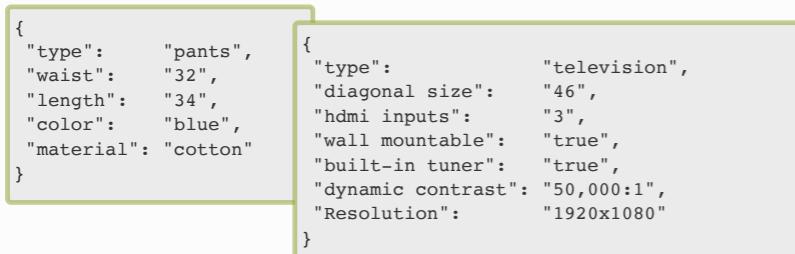


Los atributos especiales `_from` y `_to` en documentos de aristas que apuntan a otros documentos conforman un grafo en ArangoDB

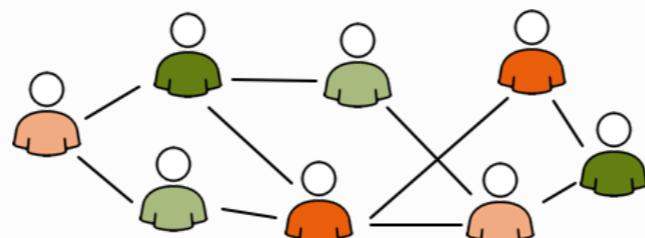


MultiModelo Nativo de ArangoDB

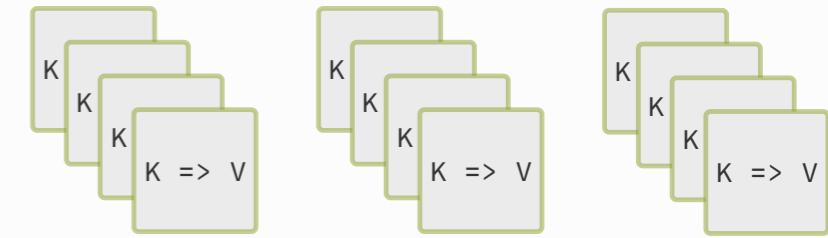
Documents - JSON



Graphs



Key Values



Implementacion Multi-Modelo

¿Cómo es implementado el multi-modelo en Arango?

- ArangoDB tiene una jerarquía de almacenamiento:
 - Puedemos crear diferentes bases de datos que pueden contener un número arbitrario de colecciones. Hay una base de datos predeterminada llamada _system.
 - Las colecciones pueden contener cantidades arbitrarias de documentos. Hay dos tipos de colección: documentos y colecciones aristas.
 - Los documentos se almacenan en formato JSON. Un documento es un objeto JSON en el nivel superior, cuyos nombres de atributo son strings y los valores pueden ser nulo, verdadero, falso, números, texto, matrices y objetos. También hay atributos del sistema (_key, _id, _rev, para aristas también _from, _to)

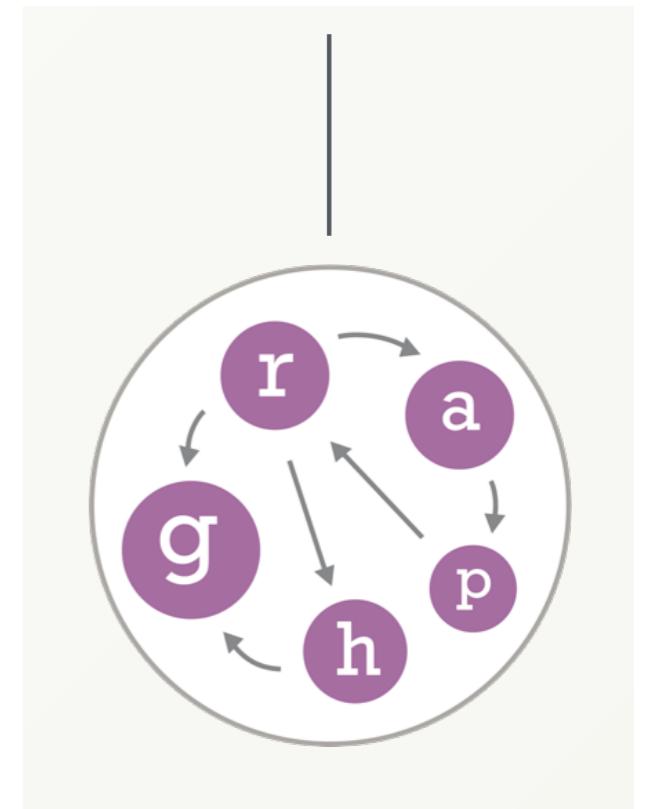


ArangoDB

Colecciones de aristas

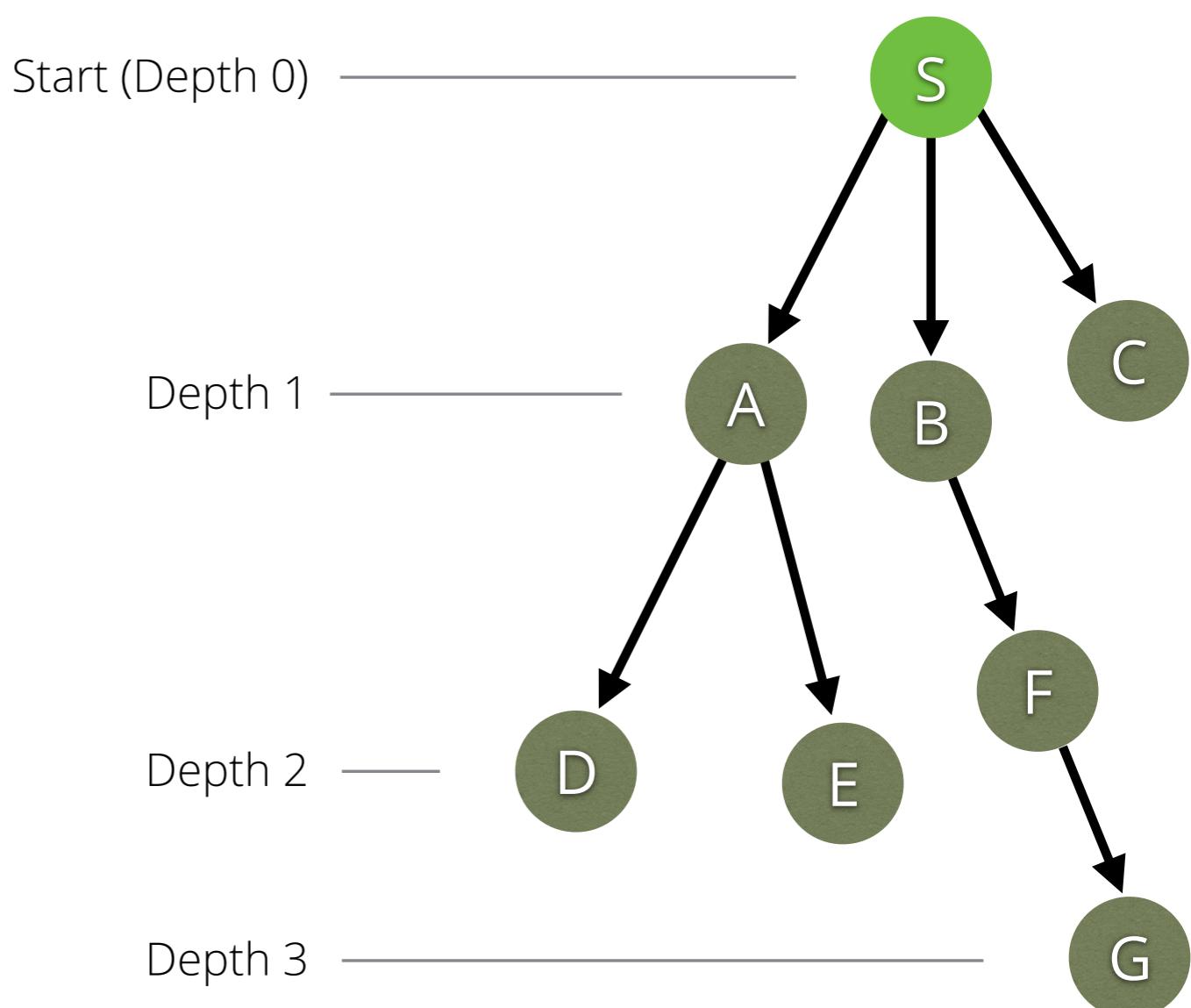
- En resumen:
- Lugar para representar/manipular relaciones
- Comparable con relaciones de N:M en sistemas SQL
- Documentos, pero con atributos especiales
 - `_from`: `_id` valor del vértice origen
 - `_to`: `_id` del vértice destino
- Índice de aristas incorporado para cada colección de aristas
- Elementos de construcción de grafos en ArangoDB.

Los atributos especiales `_from` y `_to` en documentos de aristas que apuntan a otros documentos conforman un grafo en ArangoDB



Recorridos en ArangoDB

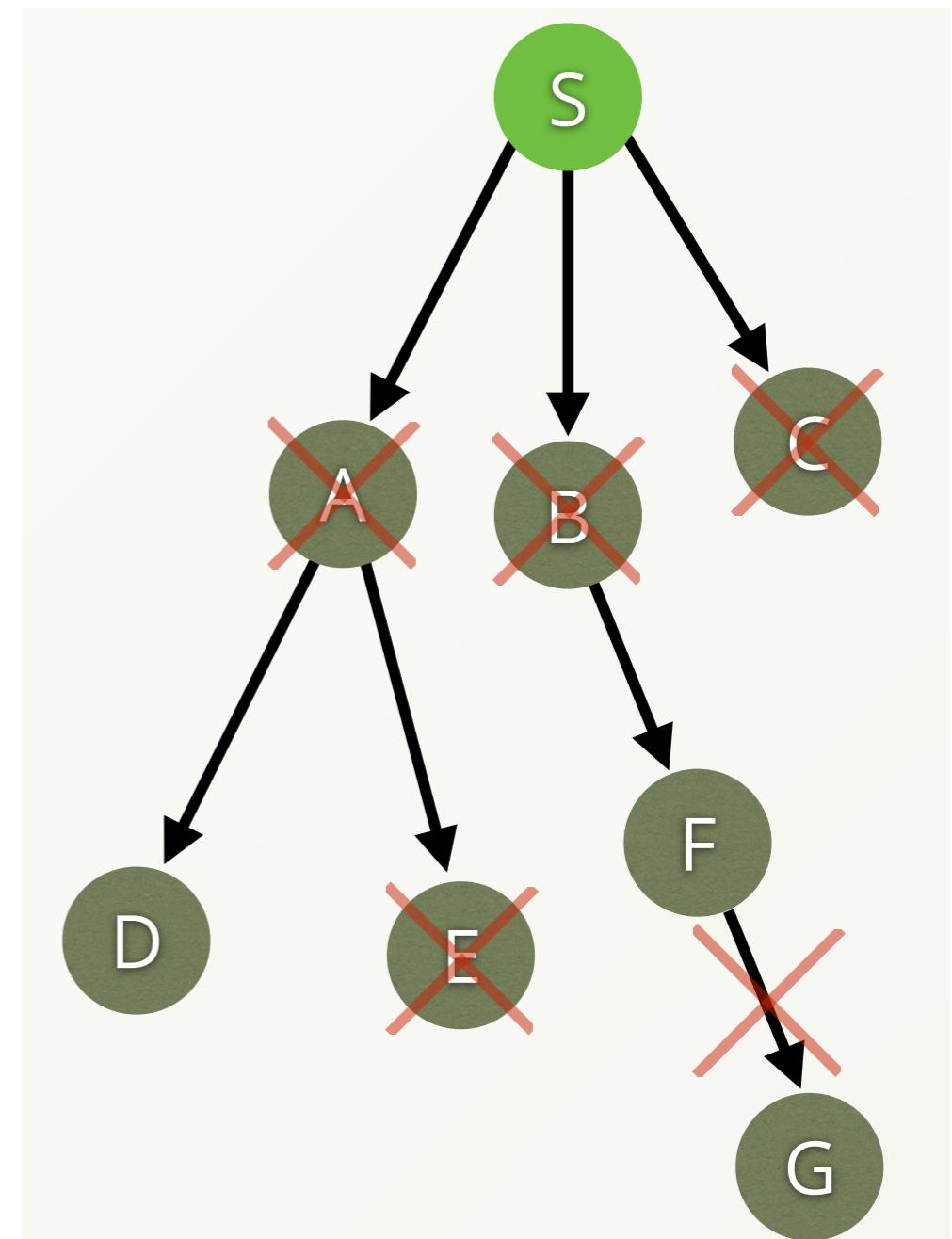
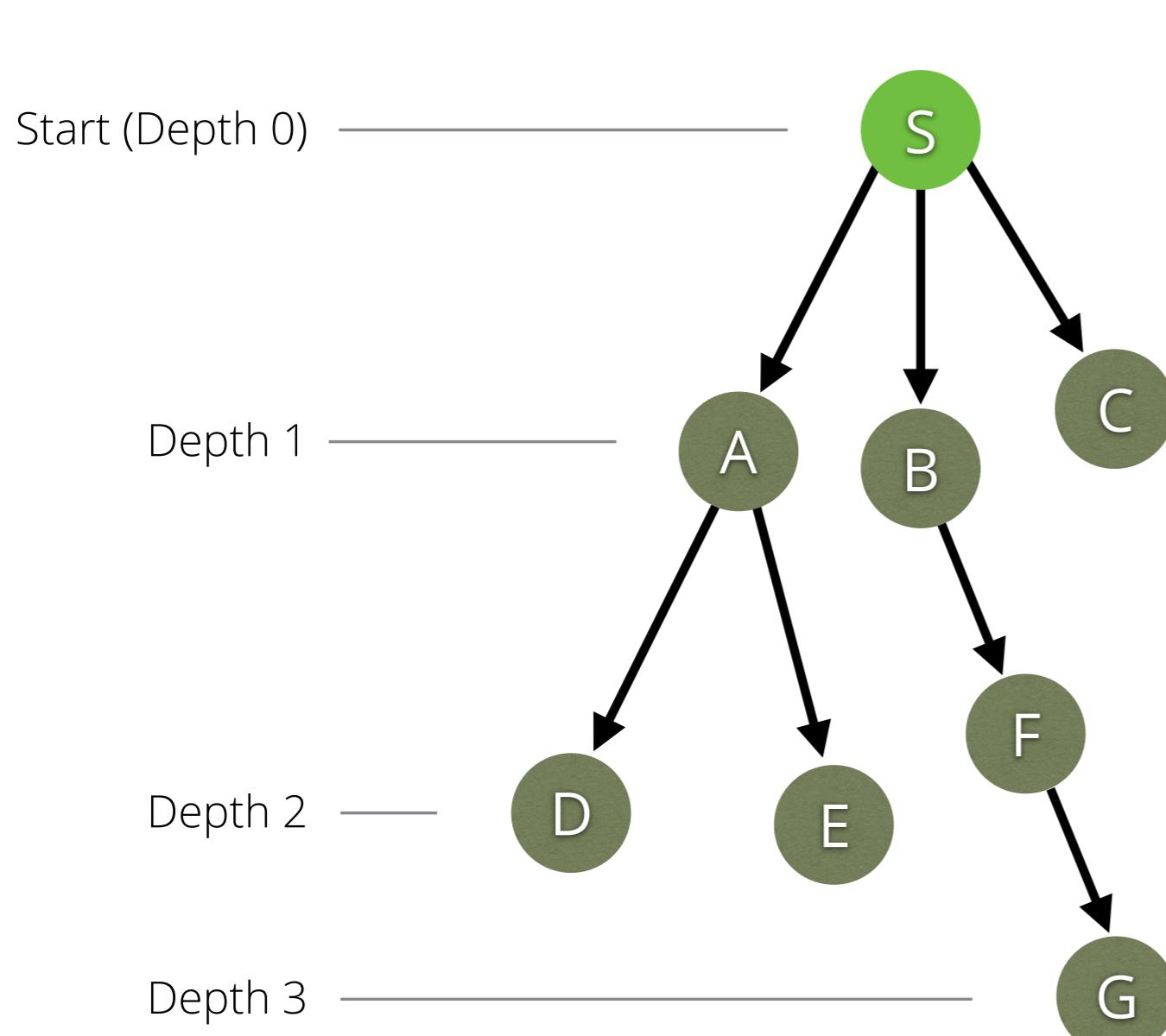
- Traversal significa recorrer a lo largo de las aristas de un grafo de ciertas maneras, opcionalmente con algunos filtros. En ArangoDB, esto se logra de manera muy eficiente mediante el índice de aristas.
- La cantidad de pasos que hay que dar en un recorrido se conoce como profundidad de recorrido.



Recorridos en ArangoDB

Ejemplos

- Un recorrido en dirección OUTBOUND con una profundidad mínima y máxima de 2.

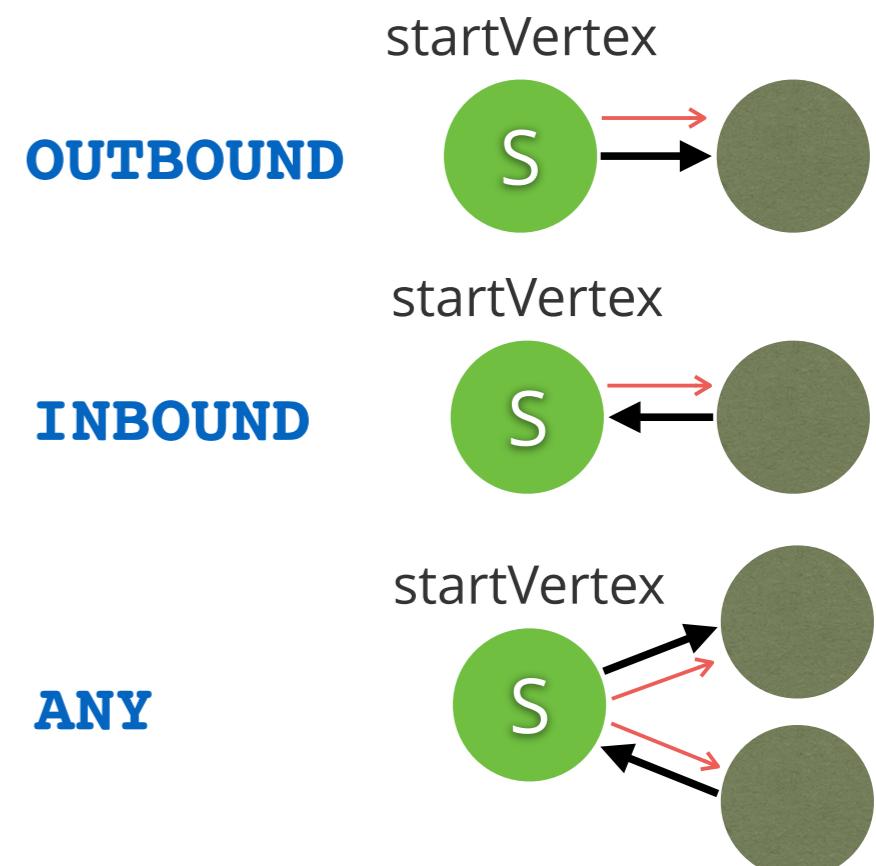


Recorridos en ArangoDB

AQL

```
FOR vertex[, edge[, path]]  
IN [min[..max]] OUTBOUND |  
INBOUND | ANY startVertex  
edgeCollection[, more...]
```

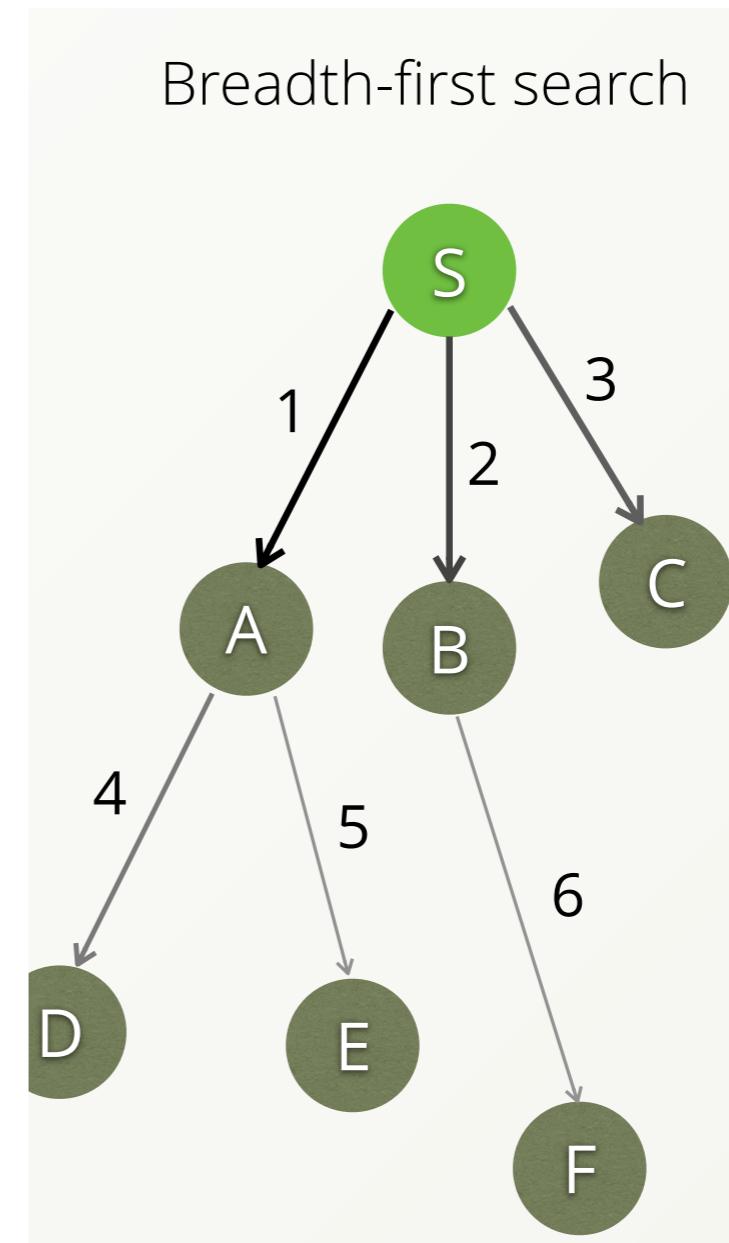
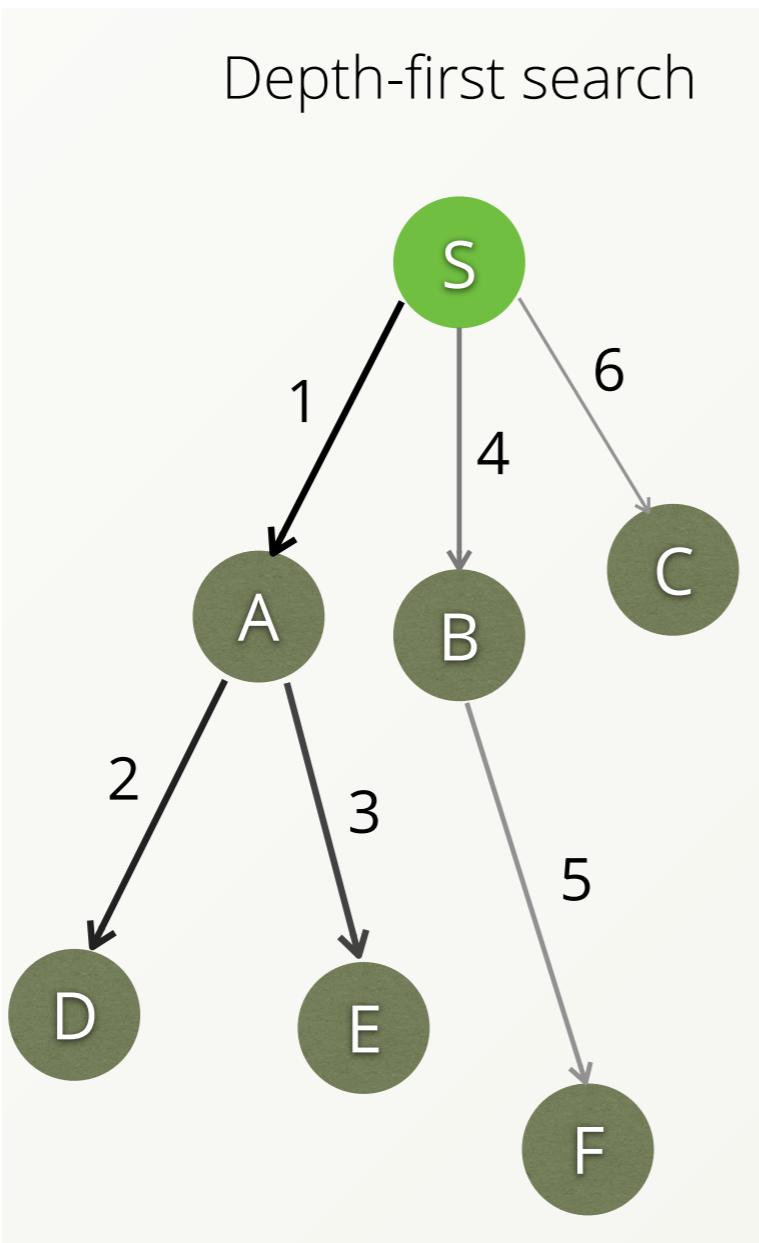
- **FOR** : vertices, aristas, caminos
- **IN** : define la profundidad minima y maxima.
- **OUTBOUND** : El recorrido sigue las aristas salientes
- **INBOUND** : El recorrido sigue las aristas salientes
- **ANY** : El recorrido sigue las aristas en cualquier dirección.



Recorridos en ArangoDB

Depth vs. Breadth-First Search

- Depth (defecto): Continúe hacia abajo por las aristas desde el vértice de inicio hasta el último vértice en ese camino o hasta alcanzar la profundidad de recorrido máxima, luego camine por los otros caminos.
- Breadth (opcional) : Siga todos las aristas desde el vértice de inicio hasta el siguiente nivel, luego siga todos las aristas de sus vecinos por otro nivel y continúe este patrón hasta que no haya más aristas para seguir o se alcance la profundidad máxima.



ArangoDB

ArangoDB Google Colab

The screenshot shows a Google Colab notebook titled "Ejemplo_ArangoDB.ipynb". The notebook contains the following content:

ArangoDB

Tutorial de Arango DB

ArangoDB es una base de datos multimodelo con modelos de datos flexibles para documentos, grafos y clave/valor. Posee el lenguaje AQL que es similar al SQL para realizar consultas.

Instalación

Antes de comenzar con ArangoDB, debemos preparar nuestro entorno y crear una base de datos temporal en el Servicio Oasis.

```
[1] %%capture
2 git clone -b oasis_connector --single-branch https://github.com/arangodb/interactive_tutorials.git
3 lsync -av interactive_tutorials/. ./ --exclude=.git
4 !pip3 install pyarango
5 !pip3 install "python-arango>=5.0"
```

```
[2] import json
2 import requests
3 import sys
4 import oasis
5 import time
6
7 from pyArango.connection import *
8 from arango import ArangoClient
```

Creamos una base de datos temporal

```
[3] 1 # Retrieve tmp credentials from ArangoDB Tutorial Service
2 login = oasis.getTempCredentials(tutorialName="AQLCrudTutorial", credentialProvider='https://tutorials.arangodb.cloud:8529/_db/_system/tutorialDB/tutorialDB')
3 # Connect to the temp database
4 conn = oasis.connect(login)
5 pyAr_db = conn[login["dbName"]]

Requesting new temp credentials.
Temp database ready to use.
```

```
[4] 1 python_arango_db = oasis.connect_python_arango(login)
2 aql = python_arango_db.aql
```

```
[5] 1 print("https://{}:{}".format(login["hostname"], login["port"]))
2 print("Username: " + login["username"])
3 print("Password: " + login["password"])
4 print("Database: " + login["dbName"])

https://tutorials.arangodb.cloud:8529
Username: TUTyldg5mspz9jvsgn79708
Password: TUTE56xixvmuxchib9htljdrv
Database: TUTy29y9plwkarpz9mkn2c628
```

Podemos usar la URL anterior para revisar nuestra base de datos arango temporal.

0s completed at 2:59 AM

File menu items: File, Edit, View, Insert, Runtime, Tools, Help. Status bar: 266302775424, RAM, Disk, Editing.

https://github.com/adigenova/uohpmd/blob/main/code/Arango_GraphDB.ipynb
<https://www.arangodb.com/docs/stable/aql/tutorial.html>

Consultas?

Consultas o comentarios?

Muchas gracias