Big-data & Cloud Storage for ML/AI

Soikanth Varma Chekuri

AppliedAICourse.com

Pre-requisites

SQL & Python

Agenda: An overview, NOT a dive-deep

1- Relational Databases : MySQL, Oracle DB, AWS RDS, ...

2. Flat-files : HDFS, Pig, Hive, Spark SQL

3. Document database: MongoDB

4. In-memory DB : Redis. Memcached

5. Inverted Indices: Elastic Search

6. Special purpose: Time-series, Graph.

Relational DB.

- 1970's - Today
- Oracle 19c, MySQL, SQL-Server, PostgreSQL, IBM DB2 ...
- Relational Algebra, SQL, Transactions, Tables
- objectives : Transaction-driven,
  Avoid duplication, save disk-space

- DS { B/B+ trees (optimize disk reads & writes)
  { Indexing (speed-up queries)

- Cloud DB vs Self-hosted DB
  - admin, scaling, cost & speed.

- Cloud DB: Aws RDS https://aws.amazon.com/rds/?c=db&sec=srv
  Azure SQL, MySQL, PostgreSQL
  https://azure.microsoft.com/en-us/product-categories/databases/
  GCP CloudSQL https://cloud.google.com/sql/

Flat - files:

- logs
- csv / tsv / JSON https://en.wikipedia.org/wiki/JSON#Example
- disb. file system : HDFS (Hadoop & Spark)
- Apache Pig: https://pig.apache.org/docs/r0.17.0/basic.html

(no - indexing ⇒ slow queries)

- Apache HIVE : HiveQL https://en.wikipedia.org/wiki/Apache_Hive
  - Indexing

- Spark SQL : SQL + Spark-programs (DataFrames)
  https://spark.apache.org/sql/
  → connect to any datasource

- Applications : 100TB-PB scale datasets
  Built on Hadoop/Spark.

Document - databases (NOSQL)

- not Table - centric, document - centric
- document : JSON, XML, ...

https://en.wikipedia.org/wiki/Document-oriented_database#Documents

https://www.mongodb.com/document-databases

- schema / fields are dynamic

Intuitive data-model:

Relational

ID	first_name	last_name	cell	city	year_of_birth	location_x	location_y
1	'Mary'	'Jones'	'516-555-2048'	'Long Island'	1986	'-73.9876'	'40.7574'

ID	user_id	profession
10	1	'Developer'
11	1	'Engineer'

ID	user_id	name	version
20	1	'MyApp'	1.0.4
21	1	'DocFinder'	2.5.7

ID	user_id	make	year
30	1	'Bentley'	1973
31	1	'Rolls Royce'	1965

MongoDB

```
{
first_name: "Mary",
last_name: "Jones",
cell: "516-555-2048",
city: "Long Island",
year_of_birth: 1986,
location: {
        type: "Point",
        coordinates: [-73.9876, 40.7574]
},
profession: ["Developer", "Engineer"],
apps: [
 { name: "MyApp",
   version: 1.0.4 },
 { name: "DocFinder",
   version: 2.5.7 }
 ],
cars: [
   { make: "Bentley",
     year: 1973 },
   { make: "Rolls Royce",
     year: 1965 }
 ]
}
```

→ fewer expensive joins

→ more disk space (cheap)

https://www.mongodb.com/document-databases

- PyMongo: https://api.mongodb.com/python/current/tutorial.html#prerequisites

- CRUD: https://docs.mongodb.com/manual/crud/

- Indexing, Scalability

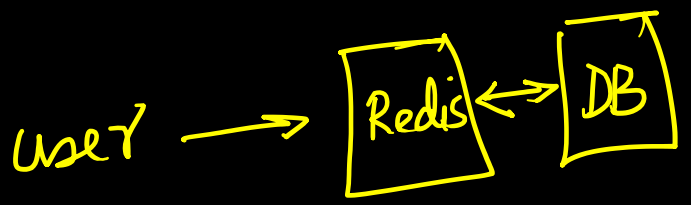
- Cloud: Aws document DB
  https://aws.amazon.com/documentdb/?c=db&sec=srv

  Azure Cosmos DB
  https://azure.microsoft.com/en-us/services/cosmos-db/

In-memory DB: Redis & Memcached

- distributed in-memory key-value datastores (hashtable/dict)
- Extremely fast read, update & writes
- Low-latency ML-applications (store features & weights)
- Real-time streaming applications

- Cache database search results



- https://redis.io/clients

- cloud: Aws Elastic cache Redis / Memcached

https://aws.amazon.com/elasticache/redis/?c=db&sec=srv

Azure Cache for Redis

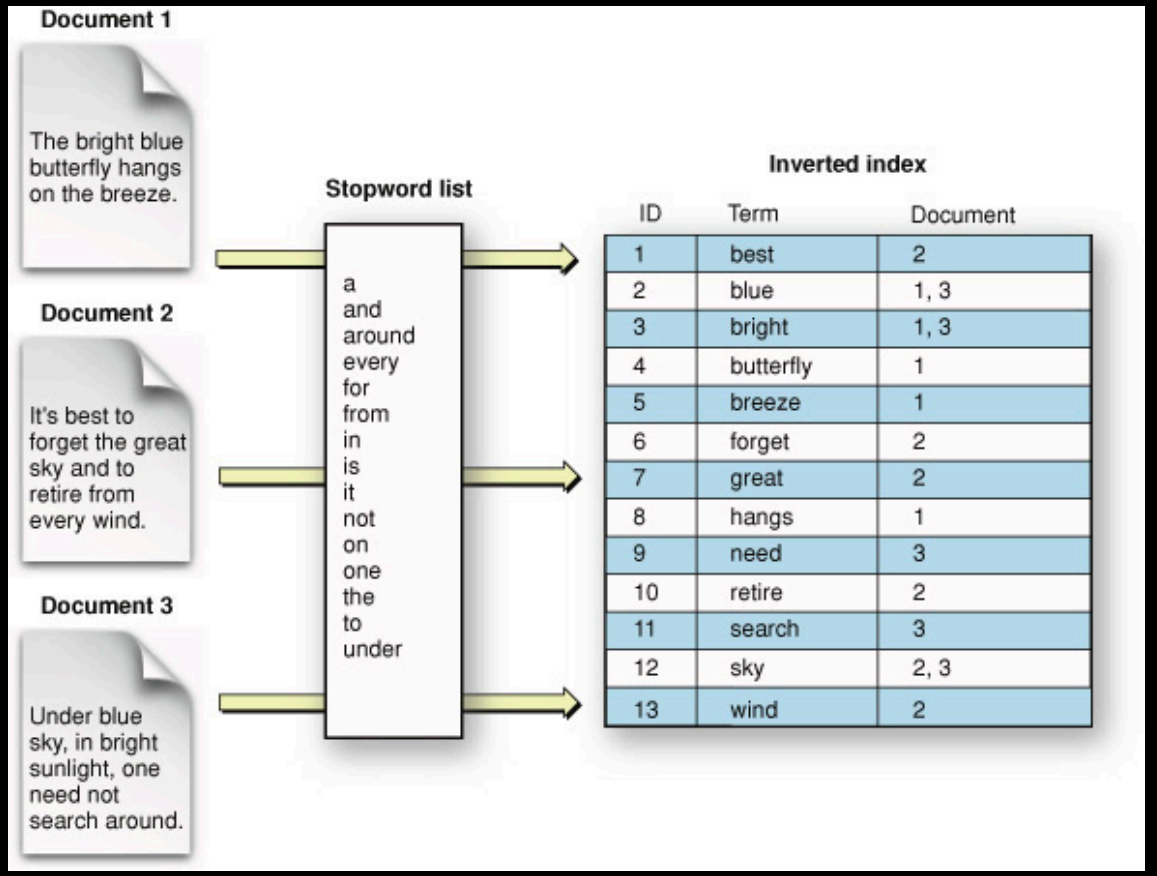
https://azure.microsoft.com/en-us/services/cache/

GCP Cloud Memorystore

https://cloud.google.com/memorystore/

Inverted Index (for Search)

eg: bright butterfly



https://community.hitachivantara.com/s/article/search-the-inverted-index

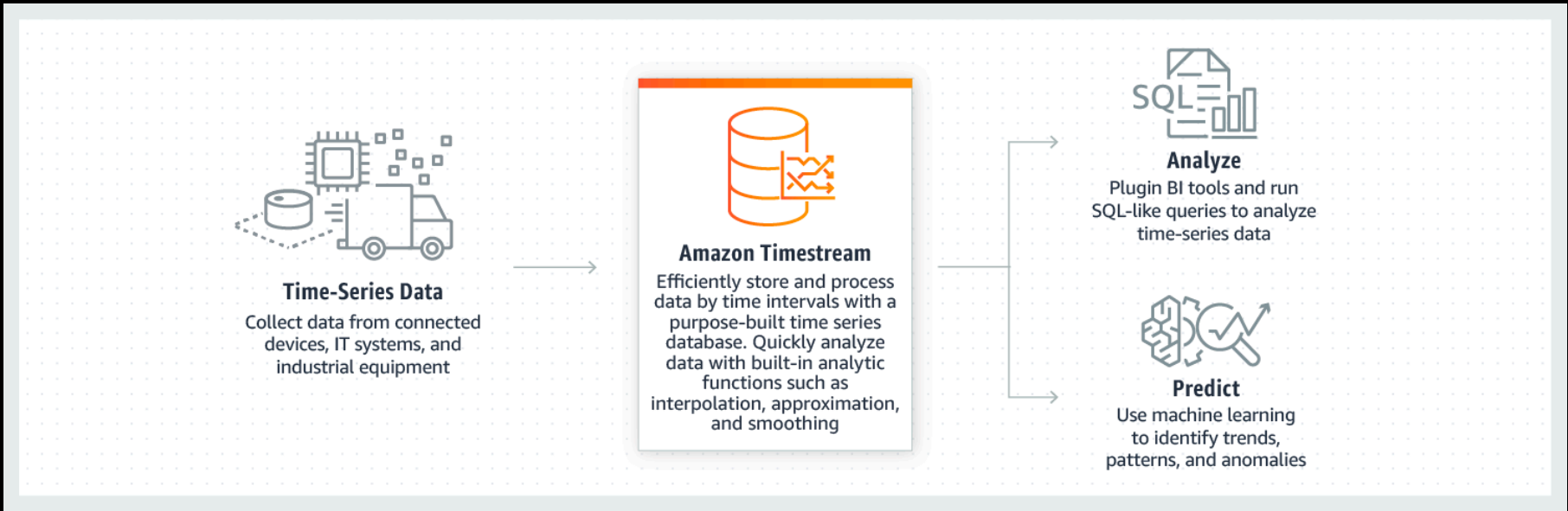
Apache Lucene, Solr

Elastic Search: https://www.elastic.co/elasticsearch

Cloud: Aws CloudSearch, Elastic Search

Time-series DB: [Aws Timestream]

- IoT applications



https://aws.amazon.com/timestream/

Graph-DB (Aws Neptune)

https://aws.amazon.com/neptune/

Social networks

Recommendation Systems

Knowledge Graph

Graph-based fraud detection