**Databases used in AI**

**Introduction:**

Essentially, an AI database is a database built with the sole purpose of speeding up [Machine Learning](https://pathmind.com/wiki/ai-vs-machine-learning-vs-deep-learning) (ML) model training. A number of tech companies are already developing dedicated AI chips to alleviate the heavy processing load in new hardware products as vendors roll out more AI-based features that require significant compute power. Artificial intelligence is understood in this context as a collection of methods that enable a computer to solve tasks that would require intelligence if they were solved by human beings.

An AI database combines [data warehousing](https://blog.datumize.com/why-dark-data-should-be-in-your-master-data-management), advanced analytics, and visualizations in an in-memory database. AI databases should be able to simultaneously ingest, explore, analyze, and visualize fast-moving, complex data within milliseconds. The goal is to lower costs, generate new revenue, and integrate ML models so that businesses can make more efficient, data-driven decisions.

**Types of Databases Used In AI:**

1. **Apache Cassandra :**

is an open-source and highly scalable NoSQL database management system that is designed to manage massive amounts of data in a faster manner. This popular database is being used by GitHub, [Netflix](https://www.slideshare.net/adrianco/migrating-netflix-from-oracle-to-global-cassandra), Instagram, Reddit, among others. Cassandra has Hadoop integration, with MapReduce support.

Advantages:

* Fault Tolerance: In Cassandra, the data is automatically replicated to multiple nodes for fault-tolerance. Also, failed nodes can be replaced with no downtime
* Elastic Scalability: Cassandra is designed with both read and write throughput, which increases linearly as new machines are added.

How Data Is stored:

* Data in Cassandra is stored as a set of rows that are organized into tables.
* Tables are also called column families.
* Each Row is identified by a primary key value.
* Data is partitioned by the primary key.
* You can get the entire data or some data based on the primary key.

1. **Couchbase:**

is an open-source, distributed, NoSQL document-oriented engagement database. It exposes a fast key-value store with managed cache for sub-millisecond data operations, purpose-built indexers for fast queries and a powerful query engine for executing SQL-like queries.

Advantages:

* Unified Programming Interface: The Couchbase Data Platform provides simple, uniform and powerful application development APIs across multiple programming languages, connectors, and tools that make building applications simple and accelerates time to market for applications.
* Big data and SQL Integrations: Couchbase Data platform includes built-in Big Data and SQL integration which allows a user to leverage tools, processing capacity, and data wherever it may reside.
* Container and Cloud Deployments: Couchbase supports all cloud platforms as well as a variety of container and virtualization technologies.

How Data Is stored:

The data is stored in a Couchbase cluster using **buckets**. Buckets are isolated, virtual containers which logically group records within a cluster. A bucket is the equivalent of a database. They provide a secure mechanism for organizing, managing and analyzing data storage.

Documents are distributed uniformly across the cluster and stored in the buckets. Buckets provide a logical grouping of physical resources within a cluster. More specifically, it is possible to configure the memory for caching data or number of replicas per each bucket.

**vBuckets**

A vBucket is defined as the owner of a subset of the key space of a Couchbase cluster. The vBucket system is used both for distributing data across the cluster and for supporting replicas on more than one node.

Every document ID belongs to a vBucket. A mapping function is used to calculate the vBucket to which a given document belongs. In Couchbase Server, that mapping function is a hashing function that takes a document ID as input and outputs a vBucket identifier. Once the vBucket identifier has been computed, a table is consulted to lookup the server that "hosts" that vBucket. The table contains one row per vBucket, pairing the vBucket to its hosting server. A server can be responsible for multiple vBuckets.

1. **DynamoDb:**

Amazon DynamoDb a fully managed, multi-region, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications. This accessible database has been using by Lyft, Airbnb, Toyota, Samsung, among others. DynamoDB offers encryption at rest which eliminates the operational burden and complexity involved in protecting sensitive data.

Advantages:

* High Availability and Durability: DynamoDB automatically spreads the data and traffic for the tables over a sufficient number of servers to handle the throughput and storage requirements while maintaining consistent as well as fast performance.
* Performance at Scale: DynamoDb provides consistent as well as single-digit millisecond response times at any scale. The DynamoDB global tables replicate the data across multiple AWS regions in order to provide fast and local access to data for globally distributed applications.

How Data Is stored:

Amazon DynamoDB stores data in partitions. A partition is an allocation of storage for a table, backed by solid state drives (SSDs) and automatically replicated across multiple Availability Zones within an AWS Region. Partition management is handled entirely by DynamoDB—you never have to manage partitions yourself.

When you create a table, the initial status of the table is CREATING. During this phase, DynamoDB allocates sufficient partitions to the table so that it can handle your provisioned throughput requirements. You can begin writing and reading table data after the table status changes to ACTIVE.

DynamoDB allocates additional partitions to a table in the following situations:

* If you increase the table's provisioned throughput settings beyond what the existing partitions can support.
* If an existing partition fills to capacity and more storage space is required.

Partition management occurs automatically in the background and is transparent to your applications. Your table remains available throughout and fully supports your provisioned throughput requirements.

1. **Elasticsearch:**

[Elasticsearch](https://www.elastic.co/) is built on Apache Lucene and is a distributed, open-source search and analytics engine for all types of data including textual, numerical, geospatial, structured and unstructured data. Elasticsearch is the central component of the Elastic Stack which is a set of open-source tools for data ingestion, enrichment, storage, analysis, and visualization.

Advantages:

* Extensive Number of Features: Besides speed, scalability and resiliency, Elasticsearch has several built-in features such as data rollups and index lifecycle management which makes efficient storing and searching data.
* Faster in Manner: Elasticsearch excels at full-text search and it is well-suited for time-sensitive use cases such as security analytics, infrastructure monitoring, etc.

How Data Is stored:

Elasticsearch is a distributed document store. Instead of storing information as rows of columnar data, Elasticsearch stores complex data structures that have been serialized as JSON documents. When you have multiple Elasticsearch nodes in a cluster, stored documents are distributed across the cluster and can be accessed immediately from any node.

When a document is stored, it is indexed and fully searchable in [near real-time](https://www.elastic.co/guide/en/elasticsearch/reference/master/near-real-time.html)--within 1 second. Elasticsearch uses a data structure called an inverted index that supports very fast full-text searches. An inverted index lists every unique word that appears in any document and identifies all of the documents each word occurs in.

An index can be thought of as an optimized collection of documents and each document is a collection of fields, which are the key-value pairs that contain your data. By default, Elasticsearch indexes all data in every field and each indexed field has a dedicated, optimized data structure. For example, text fields are stored in inverted indices, and numeric and geo fields are stored in BKD trees. The ability to use the per-field data structures to assemble and return search results is what makes Elasticsearch so fast.

Elasticsearch also has the ability to be schema-less, which means that documents can be indexed without explicitly specifying how to handle each of the different fields that might occur in a document. When dynamic mapping is enabled, Elasticsearch automatically detects and adds new fields to the index. This default behavior makes it easy to index and explore your data—​just start indexing documents and Elasticsearch will detect and map Booleans, floating point and integer values, dates, and strings to the appropriate Elasticsearch data types.

Ultimately, however, you know more about your data and how you want to use it than Elasticsearch can. You can define rules to control dynamic mapping and explicitly define mappings to take full control of how fields are stored and indexed.