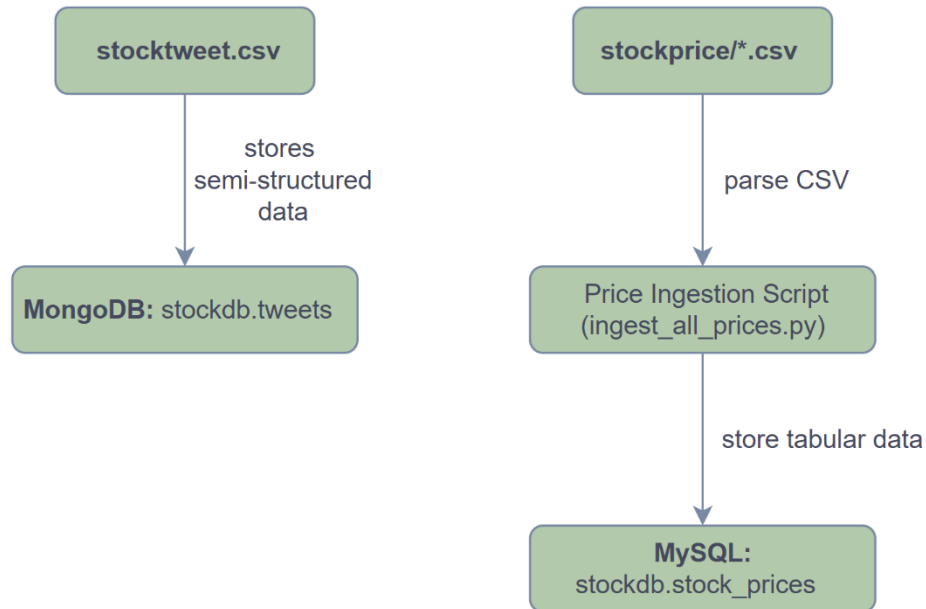


Data Collection & Storage

1. Architecture of Data Flow



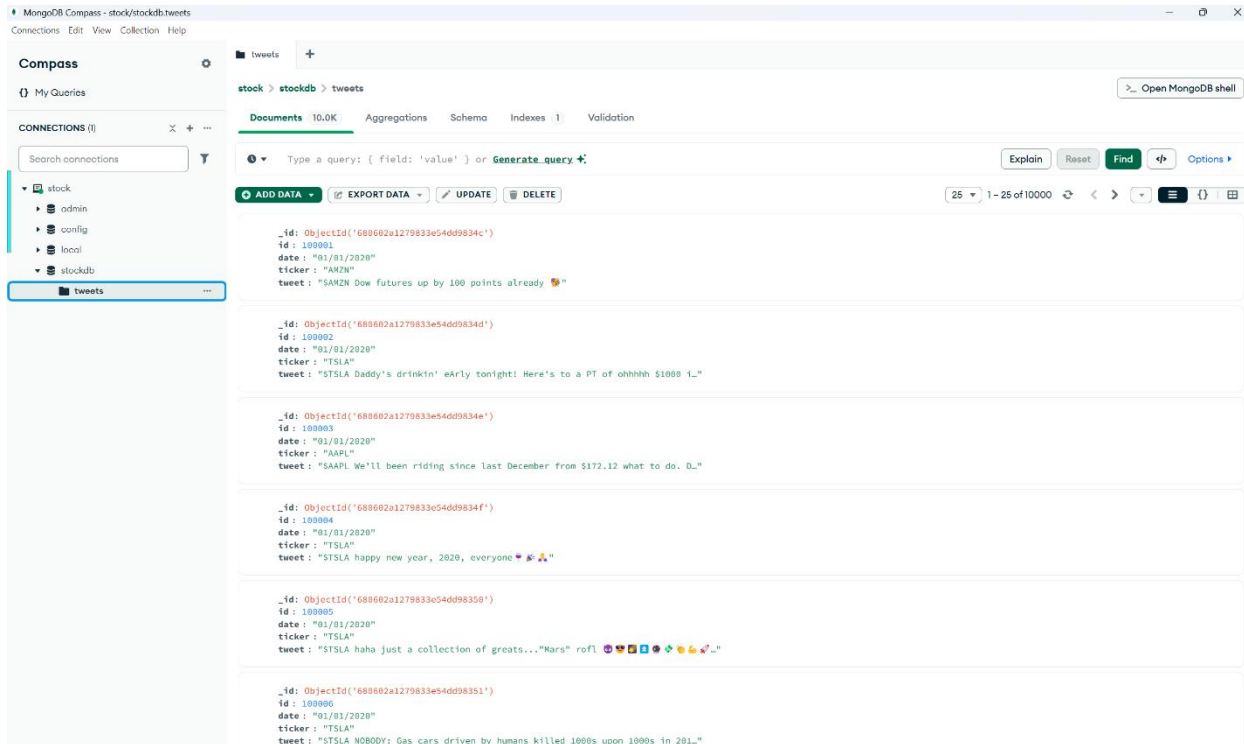
2. Tools Used

Component	Tool	Reason
NoSQL Database	MongoDB	Flexible document model, schema-on-read, native JSON storage, easy indexing on ticker & date.
Rational DB	MySQL	ACID compliance, efficient indexed time-series queries.
Scripting	Python3	Rich ecosystem like (pymongo, mysql-connector,csv), cross-platform; concise bulk loading.
GUI Validation	MongoDB Compass/ MySQL Workbench	Visual data inspection, index and schema management.

3. Storage Design Decisions

1. MongoDB (stockdb.tweets)

A) Schema:



B) Storage Engine & Configuration

- WiredTiger for document compression, high concurrency, and write throughput.
- Journaling enabled for crash recovery.

C) Scalability & Availability

- Replica Set for high availability and automatic failover.
- Sharding strategy (hash-shard on ticker) when collection grows beyond single-node capacity.

D) Retention & Archiving

- Archive older raw tweets to cold storage or S3 after sentiment extraction.

2. MySQL (stockdb.stock_prices)

A) Schema:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'stockdb' database selected. The 'Tables' pane shows the 'stock_prices' table. The 'Columns' pane for 'stock_prices' lists the following columns: id (INT), ticker (VARCHAR(10)), date (DATE), open (DECIMAL(12,4)), high (DECIMAL(12,4)), low (DECIMAL(12,4)), close (DECIMAL(12,4)), adj_close (DECIMAL(12,4)), and volume (BIGINT). The main pane shows a query: `SELECT * FROM stockdb.stock_prices`. The 'Result Grid' shows the first 14 rows of data. The 'Output' pane shows the execution log for the query.

#	Time	Action	Message	Duration / Fetch
9	13:53:25	DROP DATABASE 'stockdb';	1 row(s) affected	0.000 sec
10	13:53:36	CREATE DATABASE IF NOT EXISTS stockdb; CHARACTER SET = utf8mb4 COLLATE = utf8mb4_unicode_ci	1 row(s) affected	0.015 sec
11	13:53:41	USE stockdb;	0 row(s) affected	0.000 sec
12	13:53:49	USE stockdb;	0 row(s) affected	0.000 sec
13	13:53:53	CREATE TABLE IF NOT EXISTS stock_prices (id INT NOT NULL AUTO_INCREMENT, ticker VARCHAR(10) N...	0 row(s) affected	0.015 sec
14	13:57:37	SELECT * FROM stockdb.stock_prices LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

B) Partitioning & Maintenance

- Range Partitioning by date (e.g., monthly or yearly) to limit scan scope and speed up purging old partitions.
- Event Scheduler or external job to drop/archive partitions beyond a retention period.

C) High Availability & Scaling

- Master-Replica setup for read-scaling and zero-downtime failover.
- Evaluate MySQL Cluster or Amazon Aurora for automatic sharding and multi-AZ resilience if volumes spike.

D) Backup & Recovery

- Point-in-Time Recovery via binary logs.
- Logical Backups (mysqldump) for schema snapshots; Physical Backups (Percona XtraBackup) for large datasets.