**Batch T4**

**Practical No. 10**

**Title of Assignment : Cassandra Clustering**

**Student Name: Parshwa Herwade**

**Student PRN: 22510064**

**Title**  
Cassandra Clustering for Weather Station IoT Temperature Data

**Objective / Aim**

* To set up and configure a multi‑node Apache Cassandra cluster (group name: 2025GRP\*\*) on a single Windows machine, and then on multiple physical machines.
* To install and configure DataStax OpsCenter Community Edition to monitor and manage the Cassandra cluster.
* To design a Cassandra data model for ingesting and querying time‑series temperature data from distributed IoT weather stations.
* To demonstrate cluster operations (read/write, replication, failover) using OpsCenter.

**Introduction**  
Apache Cassandra is a highly scalable, distributed NoSQL database designed for handling large volumes of data across many commodity servers, providing high availability with no single point of failure. In IoT scenarios—such as a network of weather stations recording temperature readings every five minutes—Cassandra’s write‑optimized architecture and tunable consistency make it an ideal choice.

**Theory / Algorithms**

1. **Cassandra Architecture**
   * **Peer‑to‑Peer Nodes**: Every node in a Cassandra ring has the same role.
   * **Partitioner & Token Assignment**: Murmur3Partitioner assigns data tokens to nodes.
   * **Replication**: Data is replicated across multiple nodes according to the replication factor.
   * **Consistency Levels**: CLIENT can choose ONE, QUORUM, ALL, etc., to trade off latency vs. consistency.
2. **Gossip Protocol**
   * Used for internode communication to discover and share state information.
3. **Snitch**
   * Defines rack and data‑center awareness for replica placement.
4. **Data Modeling**
   * **Time‑Series Pattern**: Design a table with a compound primary key (weatherStationID, date, time) and clustering on time for efficient range queries.
   * **Wide Rows**: Each partition holds multiple time‑stamped temperature readings.

**Documentation: Functional Block Diagram / DFD**  
*(Insert a diagram here showing the flow: IoT Sensor → Local Collector → Cassandra Node → Replication across Cluster → OpsCenter Monitoring.)*

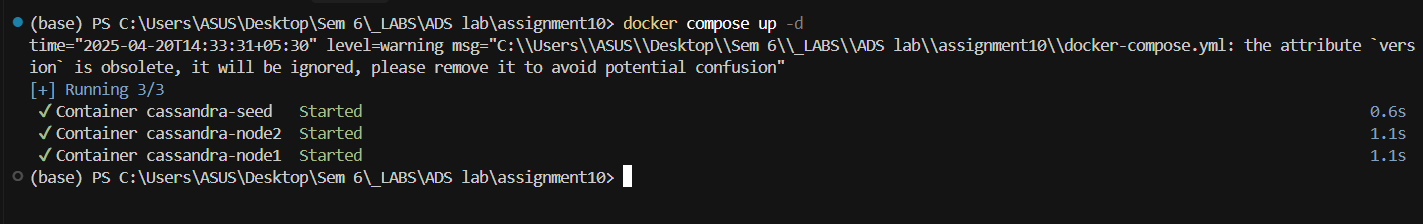
**Procedure**

1. **Single‑Machine Multi‑Node Setup**
   * Download and install Apache Cassandra (latest stable).
   * Copy the Cassandra installation directory three times as separate “nodes.”
   * Edit each node’s conf/cassandra.yaml:
     + Set cluster\_name: '2025GRP\*\*'
     + Assign unique listen\_address and storage\_port
     + Configure unique data\_file\_directories per node
     + Set ring tokens manually (initial\_token) for even data distribution
   * Start each node in its own command prompt using bin\cassandra -f.
2. **Physical Multi‑Machine Cluster**
   * Repeat above configuration on three Windows machines in the lab network.
   * Ensure all nodes can ping each other; open ports 7000, 9042, etc., in Windows Firewall.
3. **Install DataStax OpsCenter**
   * Download OpsCenter Community Edition.
   * Install on a dedicated machine or VM.
   * Point OpsCenter to the seed node’s IP and cluster name.
   * Verify cluster is discovered and basic metrics appear.

**OpsCenter Demonstration**

1. Show live metrics: throughput, latency, node status.
2. Simulate a node failure and demonstrate automatic failover and repair.

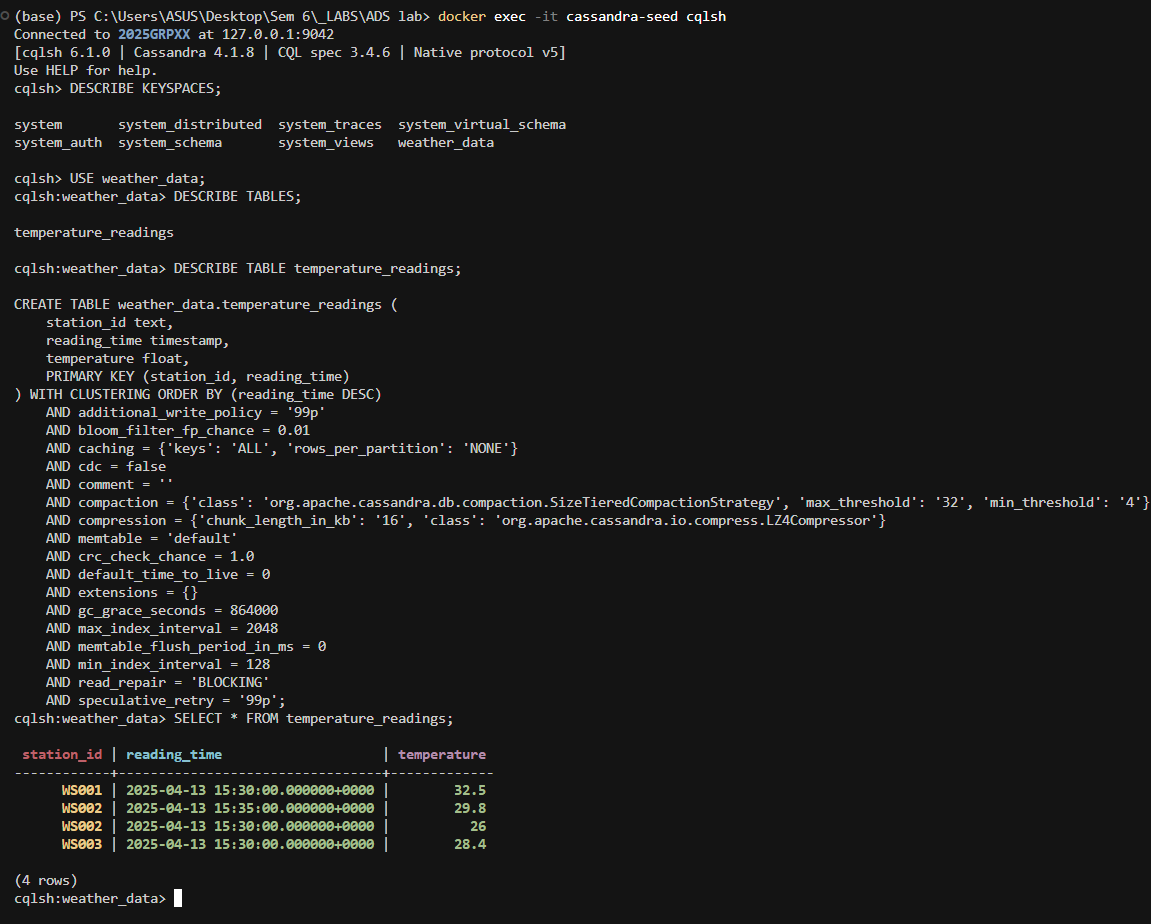
**Command : DOCKER COMPOSE UP -D**





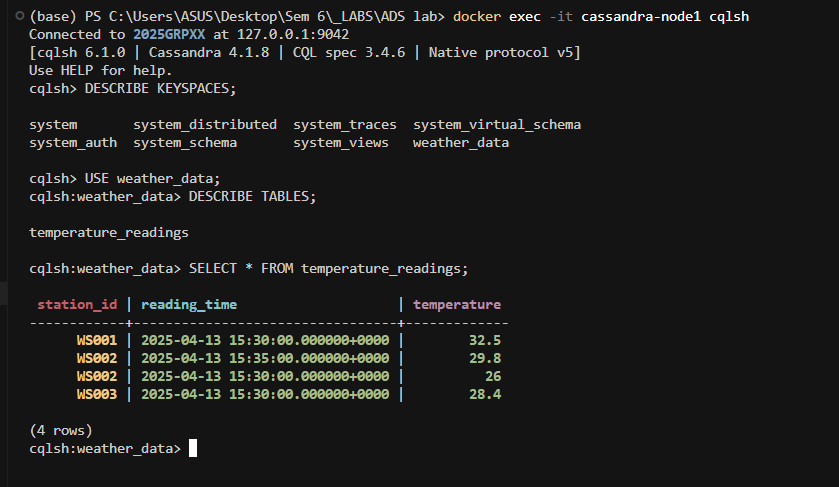
**Command : DOCKER exec -it Cassandra-seed cqlsh**

Seed Node



**Command : DOCKER exec -it Cassandra-nodel cqlsh**

Node 1:



**Command : DOCKER exec -it Cassandra-node2 cqlsh**

Node 2:

