Infrastructure Document for Real Time Streaming

1. Overview

This Docker Compose deployment sets up a complete Apache Kafka ecosystem with the following components:

- **ZooKeeper**: Coordination service for Kafka brokers.
- **Kafka Broker**: Message broker storing and serving topics.
- **Schema Registry**: Manages Avro schemas for Kafka messages.
- **Kafka Connect**: Integration framework for streaming data between Kafka and external systems.
- **Control Center**: Confluent monitoring and management UI (default **UI**).
- **Kafka UI**: Third-party web interface to browse Kafka topics and manage clusters(Optional).
- **PostgreSQL & MySQL**: Source databases for Debezium connectors and Destination Database.
- **Spark**: JupyterLab environment for data processing and analytics.

All services communicate over a dedicated Docker bridge network (`kafka-network`). Persistent data is stored on the host via mounted volumes.

---

3. Services

3.1 Zookeeper

```yaml
zookeeper:
    image: confluentinc/cp-zookeeper:6.2.15
    hostname: zookeeper
    container_name: zookeeper
    ports:
      - "2181:2181"
    volumes:
      - ./Data/zookeeper/data:/var/lib/zookeeper/data
      - ./Data/zookeeper/log:/var/lib/zookeeper/log
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_TICK_TIME: 2000
      ZOOKEEPER_DATA_DIR: /var/lib/zookeeper/data
      ZOOKEEPER_DATA_LOG_DIR: /var/lib/zookeeper/log
    networks:
      - kafka-network
```

3.2 Kafka Broker

```yaml
broker:
    image: confluentinc/cp-server:6.2.15
    hostname: broker
    container_name: broker
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
      - "29092:29092"
      - "9101:9101"
    volumes:
      - ./connectors:/connectors # source kafka connect files
      - ./Data/kafka-data:/var/lib/kafka/data # Mount a host directory for Kafka data persistence

    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: 'zookeeper:2181'
      KAFKA_ADVERTISED_LISTENERS:  PLAINTEXT_INTERNAL://broker:29092,PLAINTEXT://localhost:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT_INTERNAL:PLAINTEXT
      KAFKA_LISTENERS: PLAINTEXT_INTERNAL://0.0.0.0:29092,PLAINTEXT://0.0.0.0:9092
      KAFKA_INTER_BROKER_LISTENER_NAME: PLAINTEXT_INTERNAL
      KAFKA_METRIC_REPORTERS: io.confluent.metrics.reporter.ConfluentMetricsReporter
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_GROUP_INITIAL_REBALANCE_DELAY_MS: 0
      KAFKA_CONFLUENT_LICENSE_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_CONFLUENT_BALANCER_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_TRANSACTION_STATE_LOG_MIN_ISR: 1
      KAFKA_TRANSACTION_STATE_LOG_REPLICATION_FACTOR: 1
      KAFKA_JMX_PORT: 9101
      KAFKA_JMX_HOSTNAME: localhost
      KAFKA_CONFLUENT_SCHEMA_REGISTRY_URL: http://schema-registry:8081
      CONFLUENT_METRICS_REPORTER_BOOTSTRAP_SERVERS: broker:29092
      CONFLUENT_METRICS_REPORTER_TOPIC_REPLICAS: 1
      CONFLUENT_METRICS_ENABLE: 'true'
      CONFLUENT_SUPPORT_CUSTOMER_ID: 'anonymous'
      KAFKA_AUTO_CREATE_TOPICS_ENABLE: 'true'
      KAFKA_DEFAULT_REPLICATION_FACTOR: 1
    networks:
      - kafka-network
```

3.2 Schema Registry

```
schema-registry:
    image: confluentinc/cp-schema-registry:6.2.15
    hostname: schema-registry
    container_name: schema-registry
    depends_on:
      - broker
    ports:
      - "8081:8081"
    environment:
      SCHEMA_REGISTRY_HOST_NAME: schema-registry
      SCHEMA_REGISTRY_KAFKASTORE_BOOTSTRAP_SERVERS: 'broker:29092'
      SCHEMA_REGISTRY_LISTENERS: http://0.0.0.0:8081
    networks:
      - kafka-network
```

3.3 Kafka Connect

```
connect:
    image: confluentinc/cp-kafka-connect:6.2.15
    hostname: connect
    container_name: connect
    depends_on:
      - broker
      - schema-registry
    ports:
      - "8083:8083"
    volumes:
      - ./connectors:/home/appuser/connectors # source kafka connect files
      - ./plugins/debezium-connector/debezium-connector-mysql:/usr/share/java/debezium-connector-mysql
      - ./plugins/debezium-connector/debezium-connector-postgres:/usr/share/java/debezium-connector-postgres
      - ./plugins/kafka-connect-jdbc:/usr/share/java/kafka-connect-jdbc
    environment:
      CONNECT_BOOTSTRAP_SERVERS: 'broker:29092'
      CONNECT_REST_ADVERTISED_HOST_NAME: connect
      CONNECT_REST_PORT: 8083
      CONNECT_GROUP_ID: compose-connect-group
      CONNECT_CONFIG_STORAGE_TOPIC: docker-connect-configs
      CONNECT_CONFIG_STORAGE_REPLICATION_FACTOR: 1
      CONNECT_OFFSET_FLUSH_INTERVAL_MS: 1000
      CONNECT_OFFSET_STORAGE_TOPIC: docker-connect-offsets
      CONNECT_OFFSET_STORAGE_REPLICATION_FACTOR: 1
      CONNECT_STATUS_STORAGE_TOPIC: docker-connect-status
      CONNECT_STATUS_STORAGE_REPLICATION_FACTOR: 1
      CONNECT_KEY_CONVERTER: org.apache.kafka.connect.storage.StringConverter
      CONNECT_VALUE_CONVERTER: io.confluent.connect.avro.AvroConverter
      CONNECT_VALUE_CONVERTER_SCHEMA_REGISTRY_URL: http://schema-registry:8081
      CLASSPATH: /usr/share/java/kafka-connect-jdbc/*
      CONNECT_PRODUCER_INTERCEPTOR_CLASSES: "io.confluent.monitoring.clients.interceptor.MonitoringProducerInterceptor"
      CONNECT_CONSUMER_INTERCEPTOR_CLASSES: "io.confluent.monitoring.clients.interceptor.MonitoringConsumerInterceptor"
      CONNECT_PLUGIN_PATH: "/usr/share/java,/usr/share/java/kafka-connect-jdbc,/usr/share/confluent-hub-components"
      CONNECT_LOG4J_LOGGERS: org.apache.zookeeper=ERROR,org.I0Itec.zkclient=ERROR,org.reflections=ERROR
    networks:
      - kafka-network
```

3.4 Confluent Control Center(UI)

```
control-center:
    image: confluentinc/cp-enterprise-control-center:6.2.15
    hostname: control-center
    container_name: control-center
    depends_on:
      - broker
      - schema-registry
      - connect
    ports:
      - "9021:9021"
    environment:
      CONTROL_CENTER_CONNECT_CONNECT-DEFAULT_CLUSTER_NAME: "Dev Connect Cluster"
      CONTROL_CENTER_BOOTSTRAP_SERVERS: 'broker:29092'
      CONTROL_CENTER_CONNECT_CONNECT-DEFAULT_CLUSTER: 'http://connect:8083'
      CONTROL_CENTER_CONNECT_CLUSTER: 'http://connect:8083'
      CONTROL_CENTER_SCHEMA_REGISTRY_URL: "http://schema-registry:8081"
      CONTROL_CENTER_REPLICATION_FACTOR: 1
      CONTROL_CENTER_INTERNAL_TOPICS_PARTITIONS: 1
      CONTROL_CENTER_MONITORING_INTERCEPTOR_TOPIC_PARTITIONS: 1
      CONFLUENT_METRICS_TOPIC_REPLICATION: 1
      PORT: 9021
    networks:
      - kafka-network
```

3.5 Kafka UI (optional)

```yaml
kafka-ui:
    image: provectuslabs/kafka-ui:latest
    container_name: kafka-ui
    depends_on:
      - broker
    environment:
      KAFKA_CLUSTERS_0_NAME: local
      KAFKA_CLUSTERS_0_BOOTSTRAPSERVERS: 'broker:29092'
      KAFKA_CLUSTERS_0_ZOOKEEPER: 'zookeeper:2181'
    ports:
      - "8080:8080"
    networks:
      - kafka-network
```

3.6 PostgreSQL

```yaml
postgres:
    image: postgres:15
    container_name: postgres
    ports:
      - "5432:5432"
    environment:
      POSTGRES_USER: admin
      POSTGRES_PASSWORD: admin
      POSTGRES_DB: test_db
    volumes:
      - ./Data/postgres:/var/lib/postgresql/data
      - ./sql/init.sql:/docker-entrypoint-initdb.d/init.sql
    command:
      - "postgres"
      - "-c"
      - "wal_level=logical"
      - "-c"
      - "max_replication_slots=10"
      - "-c"
      - "max_wal_senders=10"
    healthcheck:
      test: ["CMD", "pg_isready", "-U", "demo_user"]
      interval: 10s
      timeout: 5s
      retries: 5
    networks:
      - kafka-network
```

3.7 MYSQL

```yaml
mysql:
    image: mysql:8.0
    container_name: mysql
    ports:
      - "23306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: test_db
      MYSQL_USER: admin
      MYSQL_PASSWORD: admin
    volumes:
      - ./Data/mysql:/var/lib/mysql
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "localhost"]
      interval: 10s
      timeout: 5s
      retries: 5
    networks:
      - kafka-network
```

3.8 Spark (Jupyter Notebook)

```
spark:
    build:
    context: ./docker/spark
    container_name: spark
    ports:
      - "8888:8888" # JupyterLab
      - "4040:4040" # Spark Web UI
    volumes:
      - ./Data/spark/notebooks:/home/jovyan/work
      - ./Data/spark/other_data/ivy2-cache:/home/jovyan/.ivy2
      - ./Data/spark/other_data/m2-repo:/home/jovyan/.m2/repository
      - ./plugins/kafka-connect-jdbc/mysql-connector-java-8.0.33.jar:/home/jovyan/work/jars/mysql-connector-java-8.0.33.jar
      - ./plugins/kafka-connect-jdbc/postgresql-42.7.7.jar:/home/jovyan/work/jars/postgresql-42.7.7.jar
    environment:
      - SPARK_DRIVER_MEMORY=2g
      - SPARK_EXECUTOR_MEMORY=2g
      - JUPYTER_TOKEN=spark@456
    command: >
      start-notebook.sh
      --NotebookApp.token='spark@456'
      --NotebookApp.ip='0.0.0.0'
      --NotebookApp.port=8888
      --NotebookApp.notebook_dir='/home/jovyan/work'
    networks:
      - kafka-network
```

```
Dockerfile
```

```
FROM jupyter/pyspark-notebook:latest
COPY requirements.txt /tmp/requirements.txt
RUN pip install --no-cache-dir -r /tmp/requirements.txt \
&& rm -f /tmp/requirements.txt || true
```

```
Requirement.txt
```

```
mysql-connector-python==8.3.0
Faker
ipywidgets
jupyterlab-execute-time
pyspark-stubs
```

```
fastavro
requests
streamlit
streamlit-autorefresh
```

---

4. Volumes and Persistent Data

- Host directories under `./Data/` for zookeeper, Kakfa, MySQL, PostgreSQL, and spark caches
- All the required plugin files are under `./plugins`
  - `debezium-connector-postgres`
  - `kafka-connect-jdbc-10.3.7`.jar
  - `mysql-connector-java-8.0.33`.jar
  - `postgresql-42.7.7`.jar

---

5. Scaling and Production Considerations

- **Replication:** Use replication factor > 1 for high availability
- **ZK Replacement:** Consider Kafka KRaft mode
- **Monitoring:** Integrate with Prometheus/Grafana
- **Backup:** Snapshot volumes regularly
