

# CDC Data Pipeline – Debezium + Kafka + MySQL → PostgreSQL

This document explains **how Debezium, Kafka, ZooKeeper, and your Spring Boot CDC project work together** to stream MySQL attendance data into PostgreSQL.

It also includes **all required installation commands** and the **full end-to-end flow**.

---

## 1. Architecture Overview

```
MySQL (binlog)
  ↓ (change events)
Debezium MySQL Connector
  ↓ (JSON CDC events)
Kafka Broker
  ↓ (topic: hrm_db.iclock_transaction)
Spring Boot CDC Project (Kafka Consumer)
  ↓
PostgreSQL (attendance tables)
```

### Components

Component	Purpose
MySQL	Source database generating CDC events from binlog
Debezium Connector	Captures INSERT/UPDATE/DELETE changes
Kafka	Stores events in topics
Spring Boot CDC Project	Consumes events and writes into PostgreSQL
PostgreSQL	Final attendance storage

---

## 2. Install and Start Required Services

### 2.1 Install ZooKeeper

```
sudo apt install -y zookeeperd
```

Check if running:

```
sudo systemctl status zookeeper
```

If port conflict:

```
sudo lsof -i :2181
sudo kill -9 <PID>
```

Start manually:

```
sudo /opt/kafka/bin/zookeeper-server-start.sh
/opt/kafka/config/zookeeper.properties
```

---

## 2.2 Install Kafka 3.7.1

```
cd /opt
sudo wget https://archive.apache.org/dist/kafka/3.7.1/kafka_2.13-3.7.1.tgz
sudo tar -xzf kafka_2.13-3.7.1.tgz
sudo mv kafka_2.13-3.7.1 kafka
```

Add PATH:

```
export KAFKA_HOME=/opt/kafka
export PATH=$PATH:$KAFKA_HOME/bin
source ~/.bashrc
```

Start Kafka:

```
sudo /opt/kafka/bin/kafka-server-start.sh /opt/kafka/config/server.properties
```

---

## 3. Install Debezium MySQL Connector

### Download Debezium plugin

```
sudo wget https://repo1.maven.org/maven2/io/debezium/debezium-connector-mysql/2.1.0.Final/debezium-connector-mysql-2.1.0.Final-plugin.tar.gz

sudo mkdir -p /opt/debezium/connectors
sudo tar -xzf debezium-connector-mysql-2.1.0.Final-plugin.tar.gz -C /opt/debezium/connectors
```

---

### 3.1 Start Kafka Connect Worker

```
/opt/kafka/bin/connect-distributed.sh /opt/kafka/config/connect-distributed.properties
```

---

## 4. Configure Debezium MySQL Connector

Create file:

```
/opt/kafka/config/mysql-connector.json
```

Example:

```
{
  "name": "mysql-hrm-connector",
  "config": {
    "connector.class": "io.debezium.connector.mysql.MySqlConnector",
    "database.hostname": "localhost",
    "database.port": "3306",
    "database.user": "user",
```

```
{
  "database.password": "password",
  "database.server.id": "1",
  "database.server.name": "hrm_db",
  "table.include.list": "hrm_db.iclock_transaction",
  "database.history.kafka.bootstrap.servers": "localhost:9092",
  "database.history.kafka.topic": "schema-changes.hrm_db",
  "key.converter": "org.apache.kafka.connect.json.JsonConverter",
  "value.converter": "org.apache.kafka.connect.json.JsonConverter"
}
```

## Register connector

```
curl -X POST -H "Content-Type: application/json" \
  --data @/opt/kafka/config/mysql-connector.json \
  http://localhost:8083/connectors
```

Check status:

```
curl http://localhost:8083/connectors/mysql-hrm-connector/status
```

Delete connector:

```
curl -X DELETE http://localhost:8083/connectors/mysql-hrm-connector
```

Restart connector:

```
curl -X POST http://localhost:8083/connectors/mysql-hrm-connector/restart
```

Validate config:

```
curl -X PUT -H "Content-Type: application/json" \
  --data @/opt/kafka/config/mysql-connector.json \
```

```
http://localhost:8083/connector-plugins/io.debezium.connector.mysql.MySqlConnector/config/validate
```

---

# 5. Kafka Topic Management

## List all topics

```
/opt/kafka/bin/kafka-topics.sh --bootstrap-server localhost:9092 --list
```

## Example expected topics

```
hrm_db.iclock_transaction
hrm_db.hrm_db.iclock_transaction
hrm_db.hrm_db.iclock_transaction.DLT
schema-changes.hrm_db
```

---

## 6. Consume CDC Events from Kafka

### Read initial messages:

```
/opt/kafka/bin/kafka-console-consumer.sh \  
  --bootstrap-server localhost:9092 \  
  --topic hrm_db.iclock_transaction \  
  --from-beginning \  
  --max-messages 10
```

### Read enriched topic:

```
/opt/kafka/bin/kafka-console-consumer.sh \  
  --bootstrap-server localhost:9092 \  
  --topic hrm_db.hrm_db.iclock_transaction \  
  --from-beginning \  
  --max-messages 10
```

---

## 7. Reset Kafka Consumer Group to Re-read ALL Data

When PostgreSQL data is deleted and you want **full re-sync**:

### Stop your Spring Boot CDC service

```
systemctl stop cdc-data-pipeline
```

### Reset offsets

```
/opt/kafka/bin/kafka-consumer-groups.sh \  
  --bootstrap-server localhost:9092 \  
  --group com.betopia.hrm.consumer \  
  --reset-offsets \  
  --to-earliest \  
  --execute \  
  --topic hrm_db.hrm_db.iclock_transaction
```

### Start CDC service again

```
systemctl start cdc-data-pipeline
```

---

## 8. How Debezium CDC Works

### Example MySQL row:

```
id=100, emp_code=5001, punch_time=2025-01-10 09:00
```

### Debezium produces event:

```
{
  "before": null,
  "after": { ...new row... },
  "op": "c"
}
```

	Event	Before	After	Usage
Insert	null	data	Insert attendance	
Update	before	after	Update punch time or out time	
Delete	before	null	Soft delete / hard delete	

Your **Spring Boot CDC consumer** processes events and updates PostgreSQL.

---

## 9. Spring Boot CDC Listener Logic (Summary)

### Insert Event

```
before = null
after = {...}
→ saveOrUpdateAttendance(after)
```

### Update Event

```
before != null and after != null
→ update logic
```

### Delete Event

```
before != null and after = null
→ softDelete() or hardDelete()
```

---

## 10. Full End-to-End Data Flow

1. MySQL table `iclock_transaction` changes
  2. Debezium reads binlog
  3. Writes JSON CDC message into Kafka topic
  4. Spring Boot consumes
  5. Saves/updates record in PostgreSQL
  6. PostgreSQL has up-to-date attendance data
-

# 11. Troubleshooting

## Check if ZooKeeper is running

```
sudo lsof -i :2181
```

## Check if Kafka is running

```
sudo lsof -i :9092
```

## Check if Kafka Connect is running

```
curl http://localhost:8083/
```

## Check MySQL binlog enabled

```
SHOW VARIABLES LIKE 'log_bin';
```

---

# 12. Conclusion

You now have a complete documentation showing:

- How to install Kafka + Debezium + connectors
- How CDC works from MySQL → Kafka → Spring Boot → PostgreSQL
- How to reset offsets and reload old data
- How to test connector + topics
- Clear troubleshooting steps