**I. Mô hình tổng quan**

Bước 1: Khi có sự thay đổi dữ liệu ở Data Lake (Minio)

Bước 2: Kafka nhận thông tin và khởi tạo message tương ứng

Bước 3: Flink bóc tách từ message của Kafka lấy được đường dẫn file có trên Data lake, tiến hành tải file và convert data.

Bước 4: Import file vào Iceberg theo các bảng dữ liệu tương ứng

**II. Cách hoạt động của JOB jar trên Flink**

(*Chú ý: Các dependency trong job jar và Flink job trong docker phải được đồng bộ*)

1. Kết nối Kafka

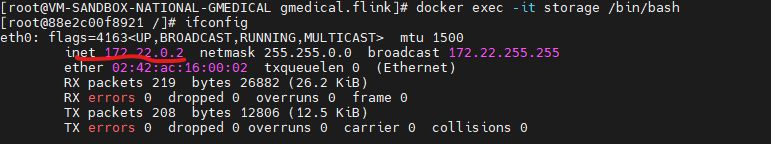
Khai báo thông tin kết nối Kafka :

Exmaple Code: //Cấu hình kết nối Kafka  
KafkaSource<String> source = KafkaSource.<String>*builder*()  
 .setBootstrapServers("10.6.8.29:9092")  
 .setTopics("test-minio")  
 .setGroupId("gmedical-id")  
 .setStartingOffsets(OffsetsInitializer.*latest*())  
 .setValueOnlyDeserializer(new SimpleStringSchema())  
 .setProperty("commit.offsets.on.checkpoint", "true")  
 .setProperty("auto.commit.interval.ms", "5000")  
 .setProperty("enable.auto.commit", "false")  
 .build();

2. Khởi tạo Catalog để quản lý các metadata trong Iceberg

Exmaple Code: *tableEnv*.executeSql(  
 "CREATE CATALOG iceberg WITH (" +  
 "'type'='iceberg'," +  
 "'catalog-impl'='org.apache.iceberg.nessie.NessieCatalog'," +  
 "'io-impl'='org.apache.iceberg.aws.s3.S3FileIO'," +  
 "'uri'='http://10.6.8.29:19120/api/v1'," +  
 "'authentication.type'='none'," +  
 "'ref'='main'," +  
 "'client.assume-role.region'='us-east-1'," +  
 "'warehouse' = 's3://warehouse'," +  
 "'s3.endpoint'='http://172.19.0.4:9000'" +  
 ")"  
);

2.1 Cách lấy s3.endpoint



3. Kiểm tra và tạo namespace(Database)

// Kiểm tra và tạo mới DATABASE theo DB  
*tableEnv*.executeSql("CREATE DATABASE IF NOT EXISTS db\_3179");

4. Kiểm tra và tạo table

// Kiểm tra và tạo mới bảng  
*tableEnv*.executeSql(TableDtos.*admissionCheckin*);

public static String *admissionCheckin* = "CREATE TABLE IF NOT EXISTS db\_3179.admision\_checkin (" + " maLk INT," + " stt INT," + " maBn STRING," + " hoTen STRING," +  
 " soCccd STRING," + " ngaySinh STRING," + " gioiTinh INT," + " nhomMau STRING," +  
 " maQuocTich STRING," + " maDanToc STRING," + " maNgheNghiep STRING," + " diaChi STRING," + " maTinhCuTru STRING," + " maHuyenCuTru STRING," +  
 " maXaCuTru STRING," + " dienThoai STRING," + " maTheBhyt STRING," + " maDkbd STRING," + " gtTheTu STRING," + " gtTheDen STRING," +  
 " ngayMienCct STRING," + " lyDoVv STRING," + " lyDoVnt STRING," + " maLyDoVnt STRING," + " chanDoanVao STRING," + " chanDoanRv STRING," +  
 " maBenhChinh STRING," + " maBenhKt STRING," + " maBenhYhct STRING," + " maPtttQt STRING," + " maDoiTuongKcb STRING," + " maNoiDi STRING," + " maNoiDen STRING," +  
 " maTaiNan STRING," + " ngayVao STRING," + " ngayVaoNoiTru STRING," + " ngayRa STRING," + " giayChuyenTuyen STRING," + " soNgayDt STRING," +  
 " ppDieuTri STRING," + " ketQuaDt INT," + " maLoaiRv STRING," + " ghiChu STRING," + " ngayTtoan STRING," + " tThuoc STRING," + " tVtyt STRING," +  
 " tTongChiBv STRING," + " tTongChiBh STRING," + " tBntt STRING," + " tBncct STRING," + " tBhtt STRING," + " tNguonKhac STRING," +  
 " tBhttGdv STRING," + " namQt INT," + " thangQt INT," + " maLoaiKcb STRING," + " maKhoa STRING," + " maCskcb STRING," + " maKhuVuc STRING," +  
 " canNang STRING," + " canNangCon STRING," + " namNamLienTuc STRING," + " ngayTaiKham STRING," + " maHsba STRING," + " maTtdv STRING," + " duPhong STRING, " + " PRIMARY KEY (maLk) NOT ENFORCED) " +  
 " WITH (\n" +  
 " 'write.metadata.delete-after-commit.enabled' = 'true',\n" +  
 " 'write.metadata.previous-versions-max' = '1',\n" +  
 " 'write.metadata.auto-merge.enabled' = 'false',\n" +  
 " 'write.parquet.compression-codec' = 'uncompressed',\n" +  
 " 'format-version' = '2',\n" +  
 " 'write.format.default' = 'parquet'\n" +  
 ")";

5. Khởi tạo dữ liệu

/Khởi tạo dữ liệu  
Table xml1Table = *tableEnv*.fromDataStream(  
 xml1Stream, *$*("maLk"), *$*("stt"), *$*("maBn"), *$*("hoTen"), *$*("soCccd"), *$*("ngaySinh")  
 , *$*("gioiTinh"), *$*("nhomMau"), *$*("maQuocTich"), *$*("maDanToc"), *$*("maNgheNghiep")  
 , *$*("diaChi"), *$*("maTinhCuTru"), *$*("maHuyenCuTru"), *$*("maXaCuTru"), *$*("dienThoai"), *$*("maTheBhyt"), *$*("maDkbd")  
 , *$*("gtTheTu"), *$*("gtTheDen"), *$*("ngayMienCct"), *$*("lyDoVv"), *$*("lyDoVnt"), *$*("maLyDoVnt"), *$*("chanDoanVao")  
 , *$*("chanDoanRv"), *$*("maBenhChinh"), *$*("maBenhKt"), *$*("maBenhYhct"), *$*("maPtttQt"), *$*("maDoiTuongKcb"), *$*("maNoiDi")  
 , *$*("maNoiDen"), *$*("maTaiNan"), *$*("ngayVao"), *$*("ngayVaoNoiTru"), *$*("ngayRa"), *$*("giayChuyenTuyen")  
 , *$*("soNgayDt"), *$*("ppDieuTri"), *$*("ketQuaDt"), *$*("maLoaiRv"), *$*("ghiChu"), *$*("ngayTtoan")  
 , *$*("tThuoc"), *$*("tVtyt"), *$*("tTongChiBv"), *$*("tTongChiBh"), *$*("tBntt"), *$*("tBncct")  
 , *$*("tBhtt"), *$*("tNguonKhac"), *$*("tBhttGdv"), *$*("namQt"), *$*("thangQt"), *$*("maLoaiKcb")  
 , *$*("maKhoa"), *$*("maCskcb"), *$*("maKhuVuc")  
 , *$*("canNang"), *$*("canNangCon"), *$*("namNamLienTuc"), *$*("ngayTaiKham"), *$*("maHsba"), *$*("maTtdv")  
 , *$*("duPhong")// đổi tên trường cho phù hợp với bảng Iceberg  
);

6. Tạo View cho table và insert data vào Iceberg

// Tạo view tạm cho Table  
*tableEnv*.createTemporaryView("xml1\_view", xml1Table);

StatementSet stmt = *tableEnv*.createStatementSet();  
App.*flinkQuery*(stmt);  
stmt.execute();

public static void flinkQuery(StatementSet stmt){

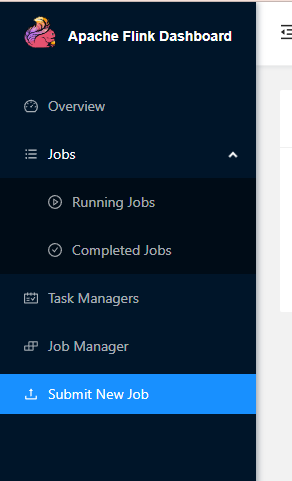
stmt.addInsertSql("INSERT INTO db\_3179.admision\_checkin SELECT \* FROM xml1\_view");

}

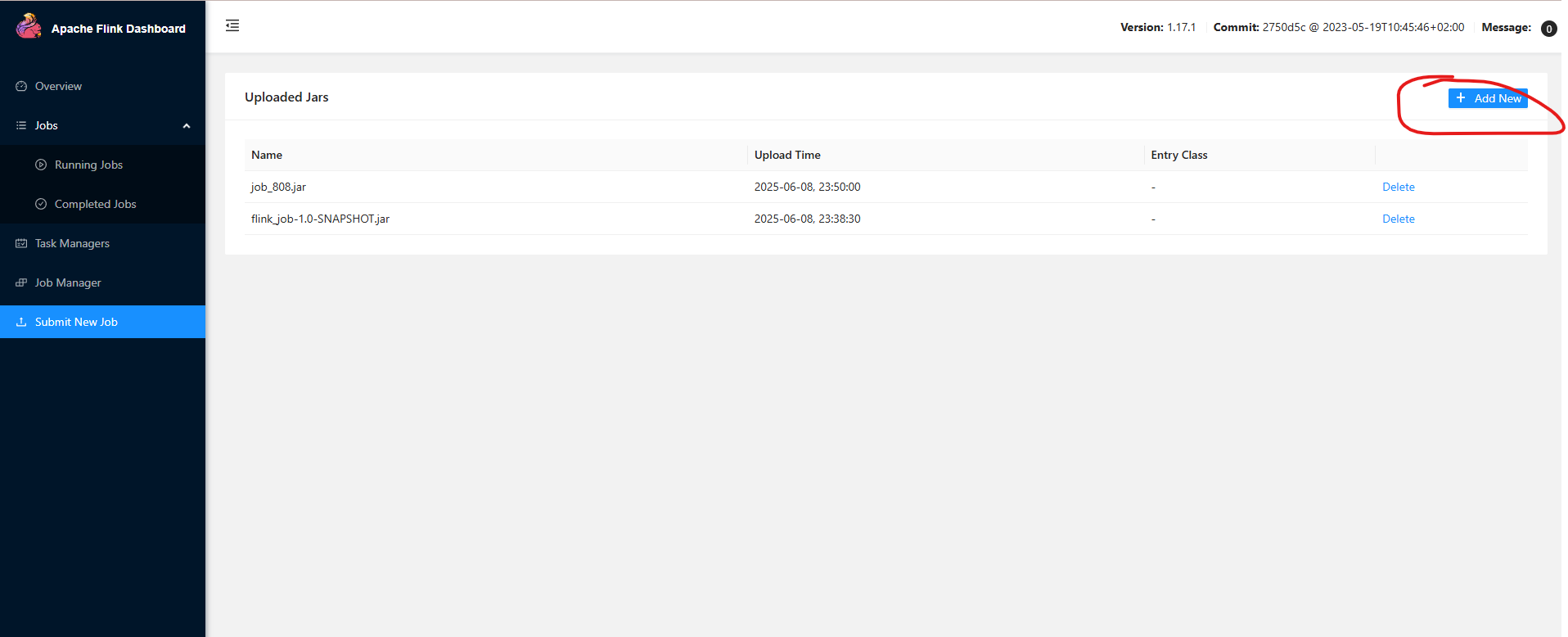
III . Chạy Job jar trên Flink UI

Bước 1: truy cập <http://10.6.8.29:8081/>

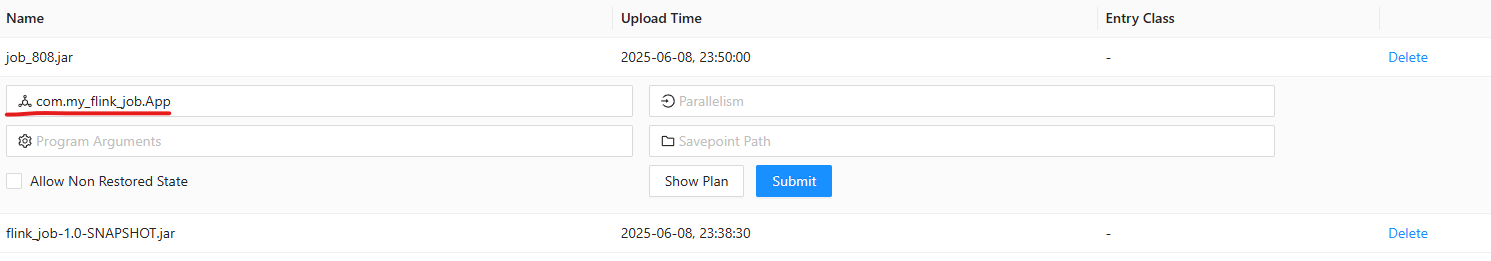
Bước 2: Chọn “Submit New Job”



Bước 3: Chọn “Add New”



Bước 4: Nhập đường dẫn đến Class cần chạy trong Job



Bước 5: Ấn “Submit”

Hình ảnh khi job chạy thành công :

