Batch operation: Bulk Operation

Batch: it used to process a large volume of data .

To use:

Add one dependency of spring batch

In @SpringBootApplication

Add @EnableBatchProcessing

And in application .propertioes

Set two properties

spring.batch.job.enabled=false  
spring.batch.jdbc.initialize-schema=always

In batch spring : it gives a reusable methods for processing large data.

There are blocks of it

Job launcher

Job

Step

Job repository

ItemReader

ItemProcesor

ItemWriter

Job repository:

To maintain job execution history and job execution step related info

Step:represents sequence of batch execution (reader + processor + writer)

ItemReader: used to read the data from source

ItemProcesser: it used to process over the data that is been read by item Reader .

ItemWriter:to write the data to the destination .

46.00

**Total Execution Flow**

**Job Launcher:**

It used to trigger the job

Usually triggered by rest api

It takes , two parameter : job and job parameters

It starts the job lifecycle

Inside job parameter : it has two parameter   
key and current timestamp in mili sec

Then there is job

**Job**

It defines the overall process

It takes job repository and step

Then start

Then build

**Step**

Its an actual job

It takes several thing as in method decleration like

Processor , reader , writer , jobRepository , platform transaction manager

To return new job , we use stepBUilder .chunk(to tell how much data we gonna insert )

Reader , processer , writer then build

**ItemReader**

it used to read input data , from resources

public FlatFileItemReader<userDet> reader(){  
 return new FlatFileItemReaderBuilder<userDet>()  
 .name("UserItemReader")  
 .resource(new ClassPathResource("data.csv"))  
 .delimited()  
 .names("id","name","email")  
 .targetType(userDet.class)  
 .build();  
}

here flatFileItemReader : this is used to read the data from file line by line .

here name , resources(to get the data actual from) , delimeted(used delimeter ) , names(columns header) , targetType(here in which class we want to merge it or not) then build

**Item Processer**

This is processer , if we want to do anything the file data , then we created the class then implements item processer which take two parameter input and output , it is an functional interface it has one abstract method that we need to implement it named as process

**Item Writer**

It used to write data into the required location like file or db

It returns jpaiTemWriter which set EntityManagerFactory

**Jpa Repository**

This is used to save a metadata about the job like track job instance , execution , parameters , and step status.

We were using pgAdmin , so we need to create some tables manually , names are

BATCH\_JOB\_INSTANCE

BATCH\_JOB\_EXECUTION

BATCH\_JOB\_EXECUTION\_CONTEXT

BATCH\_JOB\_EXECUTION\_PARAMS

BATCH\_STEP\_EXECUTION

BATCH\_STEP\_EXECUTION\_CONTEXT

And then

Create sequence for generating primary key for respected table

BATCH\_STEP\_EXECUTION\_SEQ

BATCH\_JOB\_EXECUTION\_SEQ

BATCH\_JOB\_SEQ