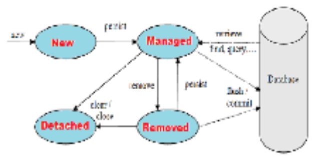
Differnet states of Entity Object in Hibernate /Spring data jpa
==



Aritist artist=new Artist(); // new state
artistRepo.save(artist); "//managed state
em.close(); // Artist becomes detached state obj

Optional<Artist> opt=artistRepo.findById(101);
Artist artist=opt.get(); // Artist is in managed state

artistRepo.deleteById(101); // Artist in removed state

em.close(); // Artist becomes detached state obj

em :: EntityManager object

=> new state :: Entity class Object is just created but not linked EntityManager
=> managed state :: Entity Class Object is representing Db table record having Synchronization
=> Deatched State :: Entity Object is not representing Db table record though record is present in Db table
=> Removed state :: The record of db table represented by Entity object is deleted.

Spring Batch (or) Spring Batch Processing (or) Spring Boot Batch

Spring Batch is given as the extension module of spring framework by pivotal team(spring team) +accenture

What is Batch Processing

=> It is the process of handling huge amount of data(chunks of data) batch by batch by reading the from one source and processing data with changes by executing logics and writing processed data to another Destination.

=>Spring batch + spring scheduling is great combination to perform batching activities in automated env.. either on "PERIOD OF TIME" or "POINT OF TIME"

Spring framework modules
---
core , aop, jdbc, orm ,mvc , jee,mail and etc.

spring extenstion modules
---
security , data jpa, batch , cloud ,scheduling, and etc..

note:: Spring boot is not a extension module .. it is an approch/methodology of spring porgramming in different modules. spring boot itself a framework having same modules of spring

usecases ::

a) Collecting CENSUS Data (population count) and storing into DB s/w category wise
the teacher collects family info on paper --> feeds to excel sheets --->uploads excel sheets to central GOVT App ---> Govt App validates the data , filters data , caterolze data ------> writes to DB s/w

b) Collecting Log Messages from ATM Machines ---> categorinizing them ----> writing to Db s/w

c) Collecting Insurance policy holders info of LIC from DB s/w -----> Indentifying the Policy matured people---> Sending that info to branches accordingly.

d) HDFC collecting Loan Holders info from DB s/w every month ------> processing and sending EMI Payment Remainder messsages/mails to customers

e) Collecting Bank Loan customers Info ------> finding Defaulters and sending notices to them

f) collecting power meter readings info ----------> calculating the bills --------> sending SMS message having the Bill info

g) Collecting Car owners info -----------> sending insurance payment renual information to customers

and etc..

While doing batch processing
=>Collect data from different sources in different formats (db table records, csv (excel data), text data and etc..)
=> Processs Data for filtering data or modifying data or categorinzing data or ordering data and etc..
=>Write data to different destinations in different fromats.. (pdf docs , csv data , db table records, xml files and etc.)

examples
=>converting DB data to Xml data after processing ✓
=>Converting one DB s/w data to another Db s/w data after procesing ✓
=> Converting JSON Data to CSV (comma seperate data)/Excel Data after processing ✓
=> Converting DB s/w data to JSON data after processing ✓
and etc... (All convertions are possible in all angels)

=> If Data of the Project is in MBs and GBs , then store data in DB s/w and manipulate it by using jdbc or spring jdbc or spring data jpa (best) or spring orm or hibernate ...
eg:: College or university Students info (for CURD Operations)

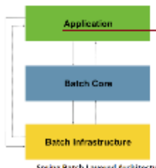
=> If Data of the Project is GBs and TBs, then store data in DB s/w and manipulate it through batch Processing (batch by batch processing) with the support of plain JDBC or spring JDBC or Spring Batch (best) (for batch by batch data processing)
eg:: Bank Data or Financial organization data or Insurance organization data

=> if the data of the Project is in TBs , PBs ,XBs,yBs and etc.. (very huge beyond storing and processing capacity of DB s/w which is BigData) then store and process that data using BigData frameworks(like hadoop or spark)
eg: Facebook , Google , twitter , youtube videos,location data ,google maps and etc..

=>In All these places we can use NoSQL DB s/w if schema/format of data is unstructed and growing dynamically.
(MongoDB, cassendra, graphQL and etc..)

note: FB kind of Apps uses multiple Storage technologies at a time in a single Project.
FB Login activities uses DB s/w
FB Posts , remainders and etc.. BigData frameworks

Spring Batch High Level Architecture



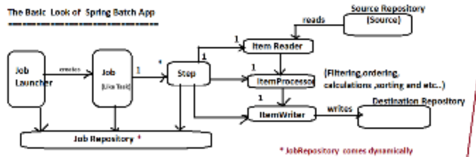
Spring Batch Layered Architecture

->Programmers code this Application

note:: In spring /spring boot batch Processing while reading /writing data from/to different resources/destinations we can use different apis/frameworks like jdbc , spring jdbc , spring data jpa ,IO streams, xml api , json api and etc..

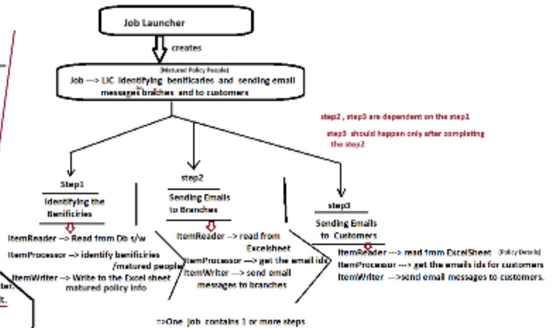
(Spring batch/spring boot batch provides abstraction on all the above said apis to simplify the programming)

The Basic Look of Spring Batch App



* JobRepository comes dynamically based on the source repository we take

=>Job Launcher launches the Job either manually or using scheduling
=> Every Job can contain 1 or more steps
=> Every Step contains 1 ItemReader , 1 ItemWriter and 1 ItemProcessor
=>Item Reader reads each item (primitive /object) from Source Repository and gives that item to Item Processor for Processing and at end of the processed items will be written to destination repository using Item Writer.
=>All activities of Job Launcher , JOB, Step will be kept tracked by JobRepository which is InMemory Repository by default.
(InMemory DB)



Job Launcher
creates
(Matured Policy People)
Job ---> LIC identifying benificaries and sending email messages to braches and to customers
Step1
Identifying the Benificiries
ItemReader --> Read from Db s/w
ItemProcessor --> identify benificiries /matured people
ItemWriter --> Write to the Excel sheet matured policy info
step2
Sending Emails to Branches
ItemReader --> read from Excelsheet
ItemProcessor ---> get the email ids
ItemWriter ---> send email messages to branches
step3
Sending Emails to Customers
ItemReader ---> read from ExcelSheet (Policy Details)
ItemProcessor ---> get the emails ids for customers
ItemWriter --->send email messages to customers.

step2 , step3 are dependent on the step1
step3 should happen only after completing the step2

=>One job contains 1 or more steps
=> Each step contains 1 ItemReader , 1 ItemWriter and 1 Item Processor

org.springframework.batch.item pkg
=> All Item Readers are implementation classes of ItemReader<T> (I)
=>All Item Processors are implementation classes of ItemProcessor<I,O> (I)
=> All Item Writers are implementation classes of ItemWriter<T> (I)

=>Generally we work with Pre-defined ItemWriter classes and pre-fined ItemReader classes becoz they are designed to deal with different types of source /Destination repositories and with different types of data formats.. But we always develop ItemProcessor manually..

->Itemreaders, ItemWriters are readily avaiable and we use user-defined ItemProcessors

->Pre-defined ItemReaders are

Reader	Purpose
FlatFileItemReader	To read data from flat files [text, csv files]
StaxEventItemReader	To read data from XML files
StoredProcedureItemReader	To read data from the stored procedures of a database
JDBCPagingItemReader	To read data from relational databases database
MongoItemReader	To read data from MongoDB
Neo4jItemReader	To read data from Neo4j DB

and etc..

AbstractCursorItemReader, AbstractItemCountingItemStreamItemReader, AbstractItemStreamItemReader, AbstractNeo4jItemReader, AbstractPaginatedDataItemReader, AbstractPagingItemReader, AmqpItemReader, AvroItemReader, FlatFileItemReader, HibernateCursorItemReader, HibernatePagingItemReader, ItemReaderAdapter, ItemStreamReader, IteratorItemReader, JdbcCursorItemReader, JdbcPagingItemReader, JmsItemReader, JpaCursorItemReader, JpaPagingItemReader, JsonItemReader, KafkaItemReader, LdifReader, ListItemReader, MappingLdifReader, MongoItemReader, MultiResourceItemReader, Neo4jItemReader, RepositoryItemReader, ResourcesItemReader, SingleItemPeekableItemReader, StaxEventItemReader, StoredProcedureItemReader, SynchronizedItemStreamReader

Excel sheet content = csv file (text content)

->Pre-defined ItemWriters are

Differnet states of Entity Object in Hibernate /Spring data jpa

persist

new

New

Managed

retrieve find, query....

Database

remove

persist

clear/ close

flush/ commit

Detached

Removed

Aritist artist=new Artist(); // new state

artistRepo.save(artist); '//managed state

em.close(); // Artist becomes

detached te obj

em :: EntityManager object

Optional<Artist> opt-artistRepo.findById(101); Artist artist=opt.get(); // Artist is in managed state artistRepo.deleteById(101); // Artist in removed state em.close(); // Artist becomes detached state obj

=> new state :: Entity class Object is just created but not linked EntityManager

=> managed state :: Entity Class Object is representing Db table record having Synchronization

=> Deatched State :: Entity Object is not representing Db table record though record is present in Db table => Removed state :: The record of db table represented by Entity object is deleted.

Spring Batch (or) Spring Batch Processing (or) Spring Boot Batch

as the

Spring Batch is given extension module of spring framework by pivotal team(spring team) +accenture ^

is

What Batch Processing

====

of

=> It is the process handling huge amount of data(chunks of data) batch by batch by reading processed the from one source and processing data with changes by executing logics and writing data to another Destination.

=>Spring batch + spring scheduling is great combination to perform batching activities in automated env.. either on "PERIOD OF TIME" or "POINT OF TIME"

Spring framework modules

core, aop, jdbc, orm,mvc, jee, mail and etc. spring extenstion modules

security, data jpa, batch, cloud,scheduling, and etc..

note:: Spring boot is not a extension module .. it is an approch/methodology

of

spring porgramming in different modules. spring boot itself a framework having same modules of spring

usecases ::

the

to

a) Collecting CENSUS Data (population count) and storing into DB s/w category wise teacher collects family info on paper --> feeds to excel sheets --->uploads excel sheets central GOVT App ---> Govt App validates the data, filters data, cateroize data ------> writes to DB s/w

b) Collecting Log Messages from ATM Machines ---> categorinizing them -----> wrting to Db s/w

info

c) Collecting Insurance policy holders of LIC from DB s/w ----->

indentifying the Policy matured people---> Sending that info to braches accordingly.

d) HDFC collecting Loan Holders Info --------> from DB s/w every month

processing and sending

EMI Payment Remainder messsages/mails to customers

to

e) Collecting Bank Loan customers Info -----> finding Defaulters and sending notices them

f) collecting power meter readings info ----------> calculating the bills --------> sending SMS message having g) Collecting Car owners info ------------> sending insurance payment

and etc..

While doing batch processing

renual information

to customers

the Bill info

=>Collect data from different sources in different formats (db table records, csv (excel data), text data and etc..) => Processs Data for filtering data or modifying data or categornizing data or ordering data and etc..

S

=>Write data to different destinations in different fromats.. (pdf docs, csv data, db table records, xml file and etc.)

examples

=>converting DB data to Xml data after processing →

=>Converting one DB s/w data to another Db s/w data after procesing

=> Converting JSON Data to CSV (comma seperate data)/Excel Data after processing

=> Converting DB s/w data to JSON data after processing →

and etc... (All convertions are possible in all angels)

=> if Data of the Project is "MBs and GBs, then store data in DB s/w and manipulate it by using jdbc or spring jdbc or spring data jpa or spring orm or hibernate ...

(best)

eg:: College or university Students Info (for CURD Operations)

=> if Data of the Project is GBs and TBs, then store data in DB s/w and manipulate it through batch Processing (batch by batch processing) with the support of plain JDBC or spring JDBC or Spring Batch (best) (For batch by batch data processing)

eg:: Bank Data or Financial organization data or Insurance organization data

in

=> if the data of the Project is TBs, PBS,XBs,yBs and etc.. (very huge beyond storing and processing capacity of DB s/w -which is BigData) then store and proccess that data using

BigData frameworks (like hadoop or spark)

eg: Facebook, Google, twitter, youtube videos, location data,goolge maps and etc..

note: FB kind of Apps uses multiple Storage technologies at a time in a single Project.

FB Login activities uses DB s/w

FB Posts, remainders and etc.. BigData

Spring Batch High Level Architecture

=====================================

frameworks

=>In All these places we can

use NoSQL DB s/w if schema/format

of data is unstructed and growing dynamically.

(MongoDB, cassendra, graphQL and etc..)

Application

Batch Core

Batch Infrastructure

Spring Batch Layered Architecture

The Basic Look of Spring Batch App

>Programmers code this Application

note:: In spring /spring boot batch Processing

while reading/writing data from/to different resources/destinations we can use different apis/frameworks like jdbc, spring jdbc, spring data jpa,IO streams,

xml api, json api and etc..

(Spring batch/spring boot batch provides abstraction on

all the above said apis to simplify the programming)

reads

Source Repository (Source)

Job Launcher

creates

(Matured Policy People)

Job ---> LIC identifying benificaries and sending email messages braches and to customers

Item Reader

1

Job

creates

Job

1

Step

Launcher

ItemProcesso

(Like Task)

1

(Filtering, ordering, calculations,sorting and etc..) Destination Repository

writes

ItemWriter

Job Repository *

Step1 Identifying the Benificiries

*** JobRepository comes dynamically**

based on the source repository we take

=>Job Launcher

launches the Job either manually or using scheduling

=> Every Job can contain 1 or more steps

=> Every Step contains 1 ItemReader, 1 ItemWriter and 1 ItemProcessor

=>Item Reader reads each Item (primitive /object) from Source Repository and gives that item

step2

Sending Emails

to Branches

ItemReader --> read from

ItemReader --> Read from Db s/w ItemProcessor --> identify benificiries /matured people/ ItemWriter --> Write to the Excel sheet matured policy info

to Item Processor for Processing and at endfocessed items will written to destination repository using Item Writer. =>All activities Job Launcher, JOB, Step will be kept tracked by JobRepository which InMemory Repository by default. org.springframework.batch.item pkg

=> All Item Readers are implementation classes itemReader<T> (1)

(InMemeory DB)

of

=>All Item Processors are implementation classes of ItemProcessor-1,0> (1)

=> All Item Writers are implementation classes of ItemWriter<< (1)

classes

=>Generally we work with Pre-defined ItemWriter classes and pre-fined ItemReader becoz they

Excelsheet

step2, step3 are dependent on the step1 step3 should happen only after completing the step2

step3

Sending Emails

to Customers

ItemReader ---> read from ExcelSheet (Policy Details)

temProcessor ---> get the email ids ItemProcessor ---> get the emails ids for customers ItemWriter --->send email messages to customers.

ItemWriter ---> send email

messages to branches

=>One job contains 1 or more steps

=> Each step contains 1 ItemReader, 1 ItemWriter and 1 Item Processor

are designed to deal with different types source /Destination repositories and with different types

of data formats.. But we always develop ItemProcessor manually.. =>Pre-defined ItemReaders are

-->Itemreaders, ItemWriters

are readily avaiable and we

use user-defined ItemProcessors

Reader

Purpose

FlatFileItemReader

To read data from flat files. (text, csv files) To read data from XML files.

StaxEventItemReader

Stored ProcedureItemReader To read data from the stored procedures of a database. To read data from relational databases database.

To read data from MongoDB.

To read data from Neo4j DB

JDBCPagingItemReader MongoItemReader Neo4jItemReader and etc.. =>Pre-defined ItemWriters are

===

```
AbstractCursorItemReader, AbstractItemCountingItemStreamItemReader,
AbstractItemStreamItemReader, AbstractNeo4jItemReader,
AbstractPaginatedDataItemReader, AbstractPagingItemReader, AmqpItemReader,
AvroItemReader, FlatFileItemReader, HibernateCursorItemReader,
HibernatePagingItemReader, ItemReaderAdapter, ItemReaderAdapter, IteratorItemReader,
JdbcCursorItemReader, JdbcPagingItemReader, JmsItemReader, JpaCursorItemReader,
JpaPagingItemReader, JsonItemReader, KafkaItemReader, LdifReader, ListItemReader,
MappingLdifReader, MongoItemReader, MultiResourceItemReader, Neo4jItemReader,
RepositoryItemReader, ResourcesItemReader, SingleItemPeekableItemReader,
StaxEventItemReader, Stored ProcedureItemReader, Synchronized ItemStreamReader
```

Excel sheet content = csv file (text content)

Writer FlatFileItemWriter StaxEventItemWriter

Purpose

To write data into flat files, csv files

To write data into XML files.

StoredProcedureItemWriter To write data into the stored procedures of a database. JDBCPagingItemWriter To write data into relational databases database.

```
AbstractFileItemWriter, AbstractItemStreamItemWriter, AmqpItemWriter, AsyncItemWriter,
AvroItemWriter, ChunkMessageChannelItemWriter, ClassifierCompositeItemWriter,
CompositeItemWriter, FlatFileItemWriter, GemfireItemWriter, HibernateItemWriter,
ItemWriterAdapter, ItemWriterAdapter, JdbcBatchItemWriter, JmsItemWriter,
JpaItemWriter, JsonFileItemWriter, KafkaItemWriter, KeyValueItemWriter,
ListItemWriter, MimeMessageItemWriter, MongoItemWriter, MultiResourceItemWriter,
Neo4jItemWriter, PropertyExtracting DelegatingItemWriter, RepositoryItemWriter,
SimpleMailMessageItemWriter, SpELMappingGemfireItemWriter, StaxEventItemWriter,
Synchronized ItemStreamWriter
```

MongoItemWriter

To write data into MongoDB.

Neo4jItemWriter

To write data into Neo4j.

and etc...

org.sf.batch.item.ItemReader<T> (I)

|----> <T> read() throws java.lang.Exception, UnexpectedInputException, ParseException,

NonTransientResourceException

org.sf.batch.item.ItemWriter<T> (I)

|--->void write (java.util.List<? extends T> items)

org.sf.batch.item.ItemProcessor<I,O> (1)

|--->O process(@NonNull 1 item) throws java.lang.Exception

=>After Processing the recived item from ItemReader

the ItemWriter can give either same object with

=>The <T> of ItemReader must

match with > of ItemProcessor and similarly the <0> of Item Processor must match with <T> of ItemWriter

note: The <I> and <0> ItemProcessor

can be same or can be different.

=>ItemReader reads one item from the source Repository

=> ItemProcessor processes the read Item given by the Reader

=> ItemWrites writes bunch of processed items to destination Repositoru based on the given batch size

modified data /unmodified data or different object

with same data or modifidata to Destination.

Case1::

=>ItemReders gets Student object from Soruce (college DB)

to

Input and output are different (Student) (Employee)

=>ItemProcessor conducts interview and converts eglible Student object Employee object => ItemWrites gets List <Employee> objs to write to Destination (Company DB)

Case2::

=> ItemReders gets PolicyHolder object from Soruce (LIC DB)

=>ItemProcessor checks payment schedule of policy holder is there in the current month or not if not there give nothing to ItemWriter otherwise gives same PolicyHolder object to ItemWriter

=> ItemWrites writes of List <PolicyHolder> objs to write to Destination (Excel sheet /another db table)

must mach

must match

Soure Repository

Item Reader<T>

ItemProcessor <1,0>

ItemWriter<i>

reads Item1 next reads Item2

process the item1 next

>

process item 2 next

Input

and

(PolicyHolder)

Output is same here (PolicyHolder)

Destination Repository

chunk size: 2 (batch size) gives_items List<Items>

writes

Developer decides the chunk size

to destination

(batch size)

(2 items)

Step flow

========

Sequence Flow of diagram of step

=>ItemReader reads 1st item from source and gives that Item to ItemProcessor for processing =>ItemReder reads 2nd item from source and gives that Item to ItemProcessor for processing

=>Once catches are processsed.. the ItemProcessor gives chunk size processed items to ItemWriter and

they written to desintation by ItemWriter..