CORSO I.F.T.S

“TECNICHE PER LA PROGETTAZIONE E LA GESTIONE DI DATABASE ”

Ing. Mariano Di Claudio

Lezione del 20/10/2014

Indice

1. **HBase e Hrider**
   - Caratteristiche chiave
   - Modello dati
   - Architettura
   - Installazione
   - Esercizio

2. **Pentaho Data Integration**
   - Aspetti teorici e pratici
   - Job e Trasformazioni
   - Esercizio

HBase

- **Hbase** è un ***NoSQL database*** di tipo ***column-oriented***, ispirato a BigTable.
- Fa parte **dell'ecosistema Hadoop** (è infatti un progetto Apache).
- Integrazione con Hadoop elevata, fa **uso di HDFS** per la ***persistenza dei dati.***

Il ***modello dati*** di Hbase si basa sui **concetti** di ***tabella*** e di ***column family***.

Hbase – Modello dati

- Una ***Tabella*** è composta da una o più ***column family***.
- Una ***column family*** solitamente contiene ***più colonne***.
- **Ogni riga** all'interno di una tabella è identificata da un ***rowId.***
- I ***valori*** sono memorizzati in ***celle***, identificate dalla tripla **(row – column - version)**. *Version è un timestamp*.
- Per definire il nome di una colonna si utilizza la seguente sintassi:

nome_column_family:nome_colonna

Hbase – Operazioni sul modello dati

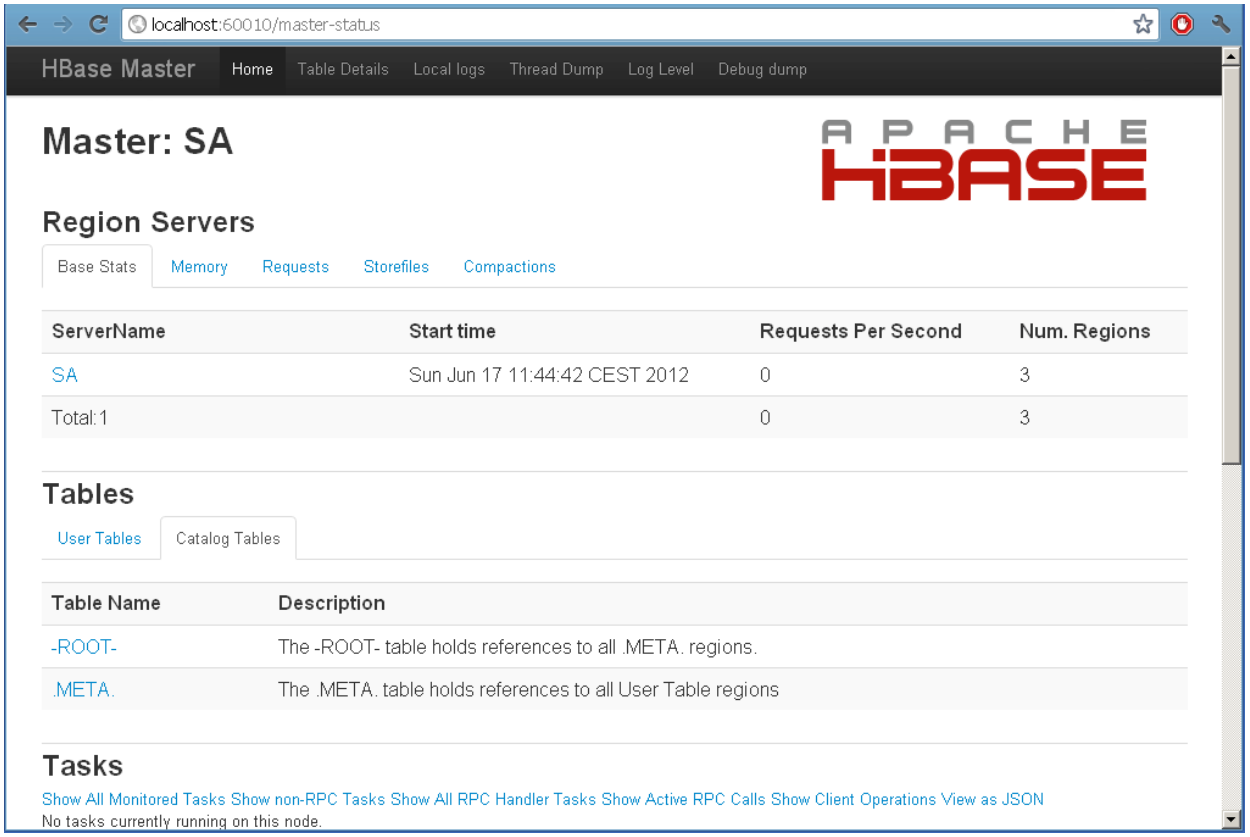
Alcune delle **possibili operazioni** eseguibili su questo **modello dati** sono:

- ***Put –*** Inserimento di una nuova riga (o aggiornamento).
- ***Get*** – estrazione dei valori da una singola riga.
- ***Scan –*** Estrazione di valori da più righe.
- ***Delete –*** Cancellazione di una riga.
- ***Disable –*** Disabilitazione di una tabella.

Hbase – Interfaccia

L'interazione con Hbase può avvenire:

- Tramite ***riga di comando*** (utilizzo della shell)
- ***Interfaccia web*** (http://ip_server_hbase:60010)



Hbase – Architettura

Hbase nasce con l'idea di implementare un **DB distribuito basato su HDFS**.

- Sul **NameNode** è presente l'istanza del ***servizio Master***.

*Il Master ha il compito di **monitorare i RegionServer** e di **gestire e controllare i cambiamenti dei metadati**.*

- Sui **DataNode** sono presenti le istanze dei ***servizi RegionServer***.

*I RegionServer gestiscono le **region**, che sono elementi che si occupano della **disponibilità e della distribuzione delle tabelle**.*

Hbase – Architettura

Hbase nasce con l'idea di implementare un **DB distribuito basato su HDFS**.

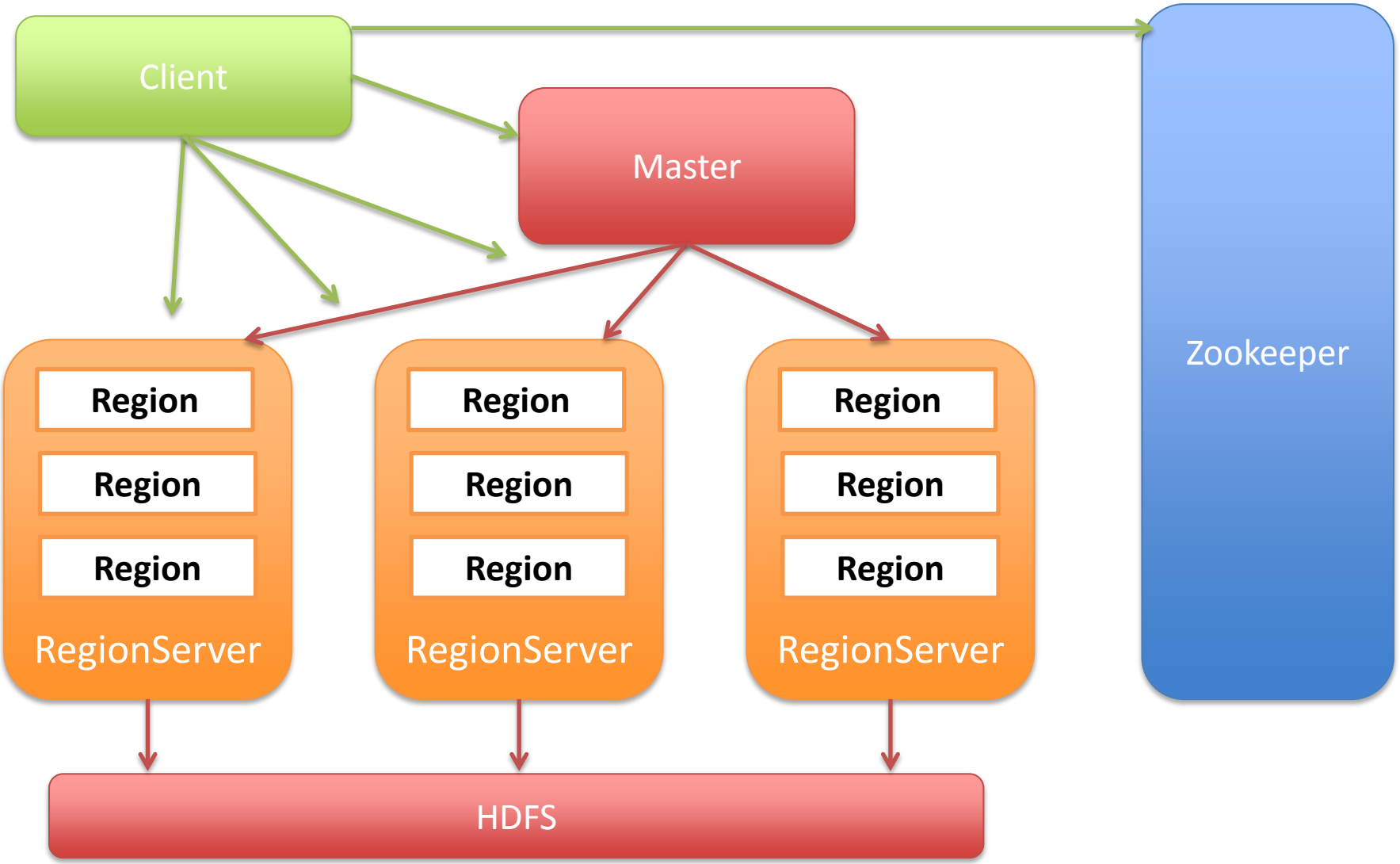
- Sul **NameNode** è presente l'istanza del ***servizio Master***.

*Il Master ha il compito di **monitorare i RegionServer** e di **gestire e controllare i cambiamenti dei metadati**.*

- Sui **DataNode** sono presenti le istanze dei ***servizi RegionServer***.

*I RegionServer gestiscono le **region**, che sono elementi che si occupano della **disponibilità e della distribuzione delle tabelle**.*

Hbase – Architettura



Hbase – Interazione

- Lo strumento principale per interagire con HBase è lo script ***hbase***, che si trova nella cartella ***bin*** dell'installazione.

- Il comando ***shell*** è quello che consente di interagire con il modello dati e di eseguire operazioni si gestione del cluster.

Comando	Descrizione	Esempio
create	Crea una tabella, specificando la colum family	*create 'tab_studenti', 'anagrafica'*
list	Restituisce l'elenco delle tabelle, si può filtrare usando delle reg exp	*list '^t'*
put	Inserisce una nuova riga	*put 'tab_studenti', 'stud1', 'anagrafica:nome', 'Gino'* *put 'tab_studenti', 'stud1', 'anagrafica:cognonome', 'Rossi'*

Hbase – Interazione

Comando	Descrizione	Esempio
scan	Restituisce un insieme di righe di una tabella	*scan 'tab_studenti', {COLUMNS => ['anagrafica:nome'], LIMIT => 10}*
get	Restituisce una riga	*get 'tab_studenti', 'stud1'*
delete	Elimina il valore di una cella	*delete 'tab_studenti', 'stud1', 'anagrafica:cognome'*
disable	Disabilita una tabella	*disable 'tab_studenti'*
drop	Elimina una tabella	*drop 'tab_studenti'*

Hbase – Installazione

- Modificare il file ***/etc/hosts*** aggiungendo le seguenti righe di codice, in modo da rimediare ad un problema di Ubuntu con le versioni 0.94.x e precedenti di HBase.

```
127.0.0.1    localhost
127.0.0.1    nome_macchina
```

- Verificare che Java sia installato

Hbase – Installazione

1. Scaricare e scompattare la versione di HBase desiderata. Nel nostro caso scompattare la cartella HBase in:

```
/home/user/bigdata/hbase
```

1. Impostare la variabile d'ambiente JAVA_HOME togliendo il commento dalla riga opportuna nel file:

```
/conf/hbase-env.sh
```

Hbase – Installazione

3. Configurare il file ***/conf/habse-site.xml*** in cui si andrà a specificare le directory sul filesystem locale in cui HBase e Zookeeper dovranno memorizzare i dati.

```
<configuration>
 <property>
   <name>hbase.rootdir</name>
   <value>file:///home/user/hbase</value>
 </property>
 <property>
   <name>hbase.zookeeper.property.dataDir</name>
   <value>/home/user/zookeeper</value>
 </property>
</configuration>
```

In questo caso saranno utilizzate le cartelle hbase e zookeeper in */home/user*, non vanno create ma lo farà Hbase in modo automatico.

Hbase – Installazione

4. Avviare Hbase con il comando ***./start-hbase.sh*** (che va lanciato una volta che si è nella cartella */bin*).

5. Verificare con il comando **jps** che i processi HMaster, HRegionServer e ZooKeepr siano avviati correttamente.

Hbase – Utilizzo da Shell

1. Connessione ad Hbase

I comandi per interfacciarsi con HBase via shell si trovano nella cartella ***bin/***. Per attivare la shell di HBase lanciare da terminale il seguente comando.

```
$ bin/ ./hbase shell
hbase(main):001:0>
```

Hbase – Utilizzo da Shell

2. Creazione di una Tabella

Per la creazione di una tabella va utilizzato il comando ***create***, e va poi specificato il nome della tabella e della ColumnFamily.

```
$ hbase> create 'persone', 'dati'
0 row(s) in 1.2200 seconds
```

Hbase – Utilizzo da Shell

3. Informazioni su una Tabella

Per ottenere informazioni su una tabella utilizzare il comando ***list.*** Se non si specifica il nome della tabella si avrà l'elenco di tutte le tabelle presenti nel DB.

```
hbase> list 'persone'
TABLE
persone
1 row(s) in 0.0350 seconds
```

Hbase – Utilizzo da Shell

4. Popolare la Tabella

Per inserire valori all'interno di una tabella utilizzare il comando ***put***

```
hbase> put 'persone', 'row1', 'dati:nome', 'gino'
0 row(s) in 0.1770 seconds

hbase> put 'persone', 'row1', 'dati:cognome', 'rossi'
0 row(s) in 0.0160 seconds

hbase> put 'persone', 'row2', 'dati:cognome', 'verdi'
0 row(s) in 0.0260 seconds

hbase> put 'persone', 'row2', 'dati:professione', 'musicista'
0 row(s) in 0.0350 seconds
```

Hbase – Utilizzo da Shell

5. Scansione della Tabella

La scansione di una tabella può essere fatta mediante il comando ***scan.*** Si può decidere si limitare il numero di risultati da visualizzare oppure ottenere la visualizzazione completa.

```
hbase> scan 'persone'
 ROW          COLUMN+CELL
 row1         column=dati:nome, timestamp=140375941, value=gino
 row1         column=dati:cognome, timestamp=140375943, value=rossi
 row2         column=dati:cognome, timestamp=140375950, value=verdi
 row2         column=dati:professione, timestamp=140375950, value=musicista
 4 row(s) in 0.0440 seconds
```

Hbase – Utilizzo da Shell

6. Ottenere i dati di una singola riga

In questo caso utilizzare il comando ***get***, specificando oltre al nome della tabella anche quello della riga.

```
hbase> get 'persone', 'row1'
COLUMN                          CELL
dati:nome                       timestamp=1403759475114, value=gino
dati:cognome                    timestamp=1403759475218, value=rossi
2 row(s) in 0.0230 seconds
```

Hbase – Utilizzo da Shell

7. Disabilitare e cancellare una tabella

Per disabilitare una tabella utilizzare il comando ***disable***, per cancellarla utilizzare il comando ***drop***.

```
hbase> disable 'persone'
0 row(s) in 1.6270 seconds

hbase> drop 'persone'
0 row(s) in 0.2900 seconds
```

8. Per uscire dalla shell HBase utilizzare il comando *quit*

Pentaho Data Integration (PDI)

- **Pentaho** è un framework che contiene diversi pacchetti tra loro integrati che consentono la gestione completa:

- ***Problematiche della Business Intelligence***
- ***Problematiche dei Data Warehouse.***
- ***Problematiche legate ai Big Data.***

- **Kettle** e' la ***componente ETL*** di Pentaho, ovvero si occupa del trasferimento e della trasformazione dei dati.

Pentaho Data Integration (PDI)

ETL (*Extract, Transform e Load*)

- E' un processo utilizzato per lavorare con database e data warehouse che si compone di tre fasi.

1. ***Estrazione*** *dei dati dalle sorgenti di informazioni.*

2. ***Trasformazione*** *dei dati per adattarli alle esigenze operazionali e migliorarne la qualità.*

3. ***Caricamento*** *dei dati in sistemi di sintesi (database operazionali, repository, data warehouse o data mart…)*

I **tool ETL** sono molto utili **per preparare i dati** raccolti alla successiva **fase di analisi** ed eventuale traduzione in RDF.

Kettle – Caratteristiche principali

- **Kettle è sviluppato in Java**, garantisce quindi la *compatibilità e portabilità con i principali sistemi operativi* (Windows, Linux, OS X..).

- E' disponibile sia in **versione open source** che **enterprise**.

- Offre la possibilità di **interfacciarsi con i principali Database di tipo NoSQL** (Hbase, Cassandra, MongoDB, CouchDB…).

- Sviluppo e configurazione di procedure realizzato tramite **interfaccia grafica con un approccio drag&drop** delle componenti necessarie.

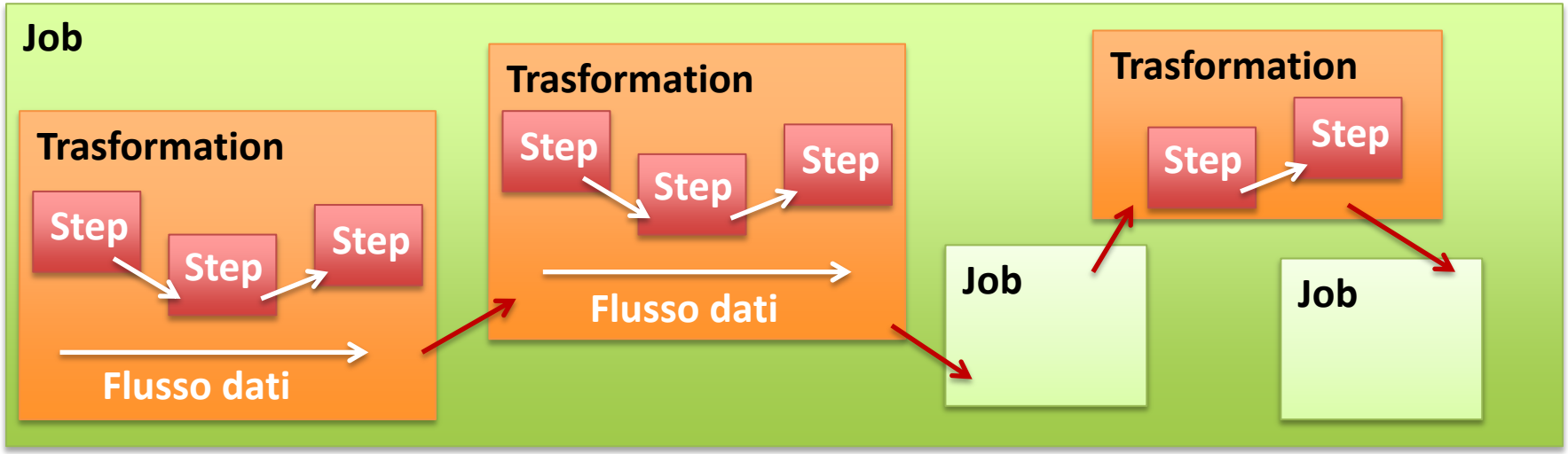
(-) *Kettle non è in grado di trasformare i dati elaborati in triple RDF*, il che implica la necessità di utilizzare altri strumenti in una fase successiva come ad esempio Karma.

Kettle – Dettagli

- Dal punto di vista del funzionamento Kettle è basato sui concetti chiave:
  - **Job** (file con estensione .kjb)
  - **Trasformazione** (con estensione .ktr), composta da diversi ***step***.
- I componenti principali di Kettle sono:
  - **Spoon** – Editor grafico per la modellazione dei processi ETL.
  - **Pan** – Esecuzione da riga di comando delle trasformazioni.
  - **Kitchen** – Esecuzione da riga di comando dei Job.

Kettle – Struttura operativa

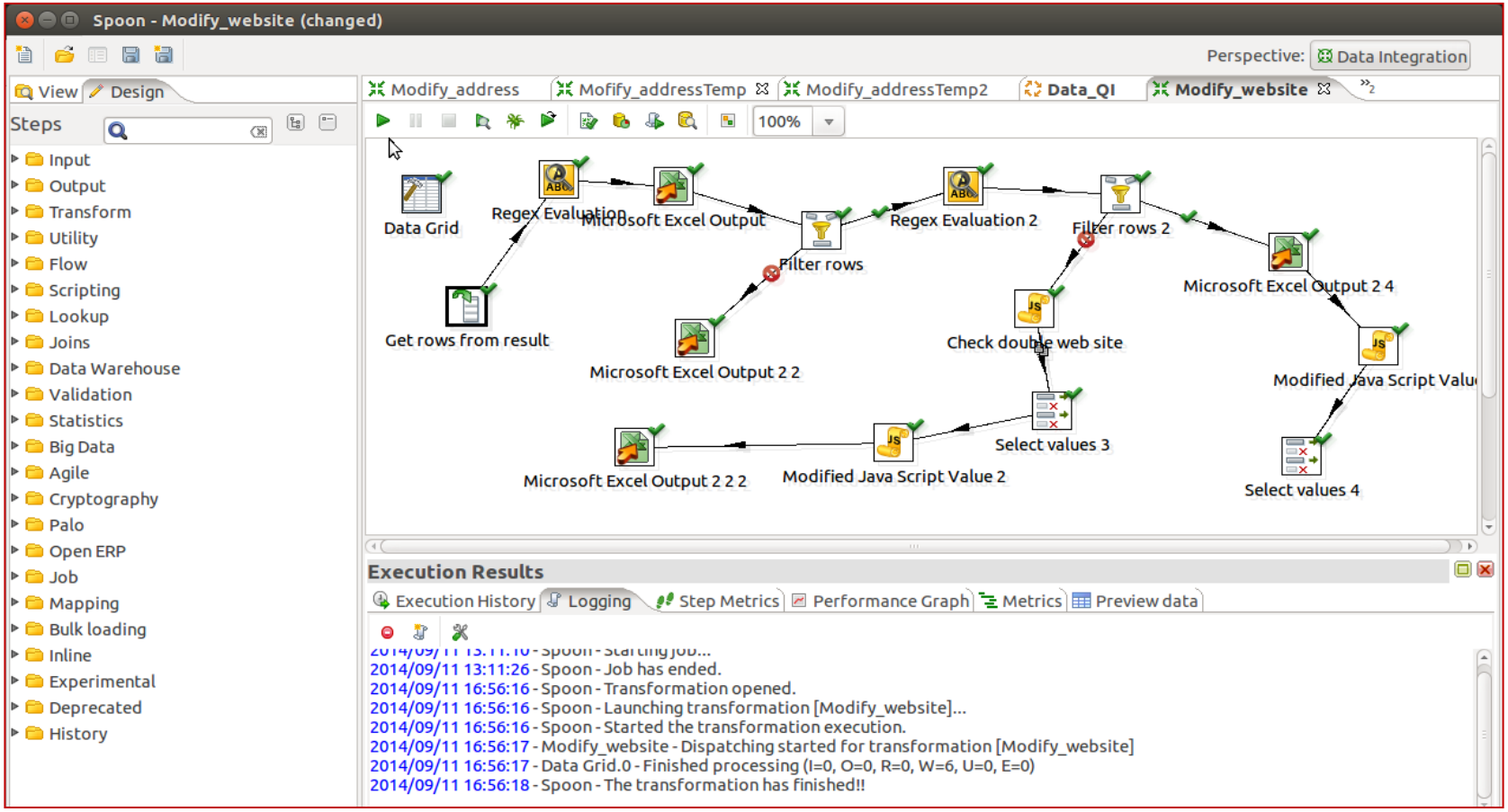
I **componenti** operativi di **Kettle** sono **organizzati** nel seguente modo:



- I **dati sono visti come flusso di righe** da uno step all'altro.
- Gli **step** sono eseguiti **parallelamente in thread separati**, *non c'è obbligatoriamente un punto di inizio o di fine di una trasformazione*.
- I **job gestiscono l'esecuzione sequenziale** delle entità di livello inferiore, *trasformazioni o altri job*.

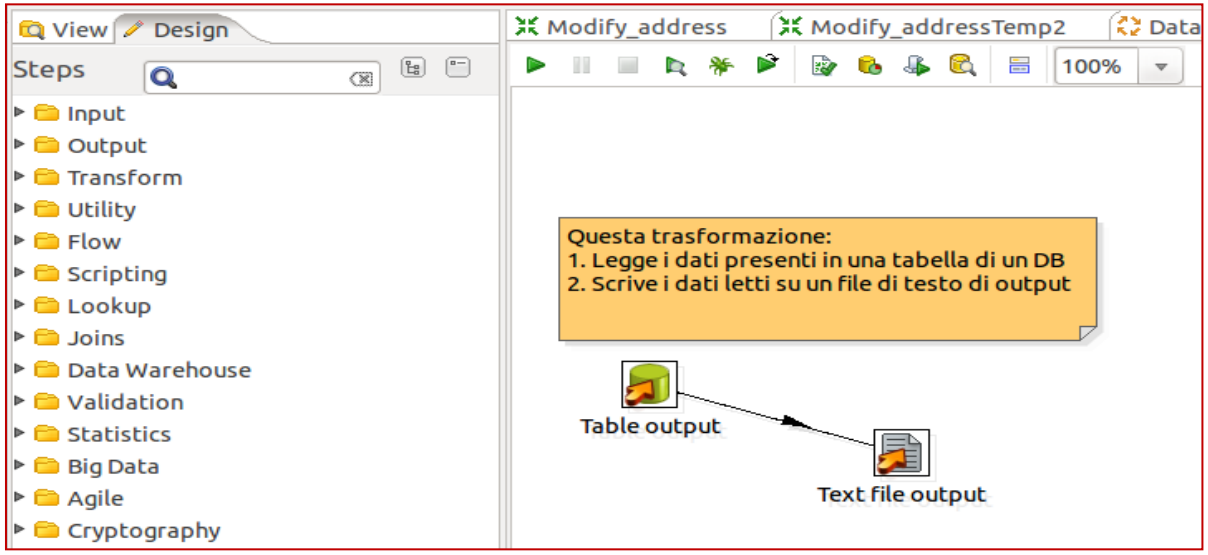
Kettle – Spoon

Per **avviare Spoon**, basta lanciare da riga di comando l'istruzione ***./spoon.sh***



Kettle – Trasformazioni

- Le ***Trasformazioni*** definiscono come i dati devono essere **raccolti**, **trasformati** e **ricaricati**.
- Sono *composte da una serie* di ***Step***, connessi tra loro mediante *collegamenti chiamati* ***Hop.***
- Tipicamente una trasformazione presenta uno ***Step di input***, uno o più ***Step di trasformazione*** e uno o più ***Step di output***.



Kettle – Trasformazioni

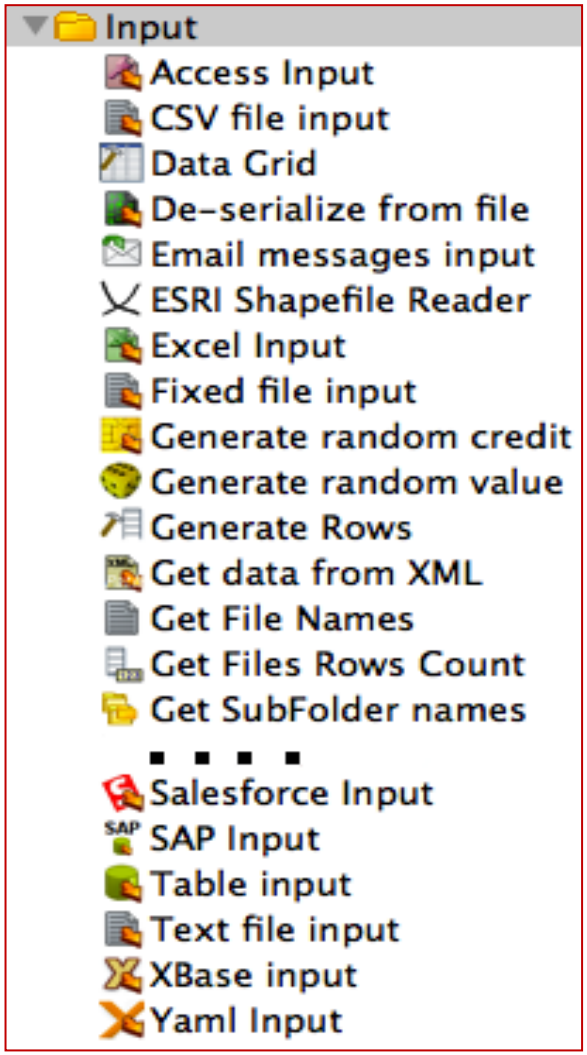
- Ci sono **diversi possibili Step**, organizzati per tipologia: ***Input, Output, Utility, Scripting, Flow, Validation, Lookup, Statistics***…etc.

- Ogni tipologia di Step offre a sua volta decine di possibilità.

- Ad esempio uno Step di input può prelevare i dati da sorgenti differenti:
  - ***Da una tabella di un DB relazionale.***
  - ***Da un file CSV.***
  - ***Da un foglio MS-Excel.***

- Gli **Hop** *che collegano i vari* **Step** **rappresentano il flusso dei dati** e trasportano il contenuto dei Field da uno step a quello successivo, *ma non definiscono la sequenza di esecuzione*.

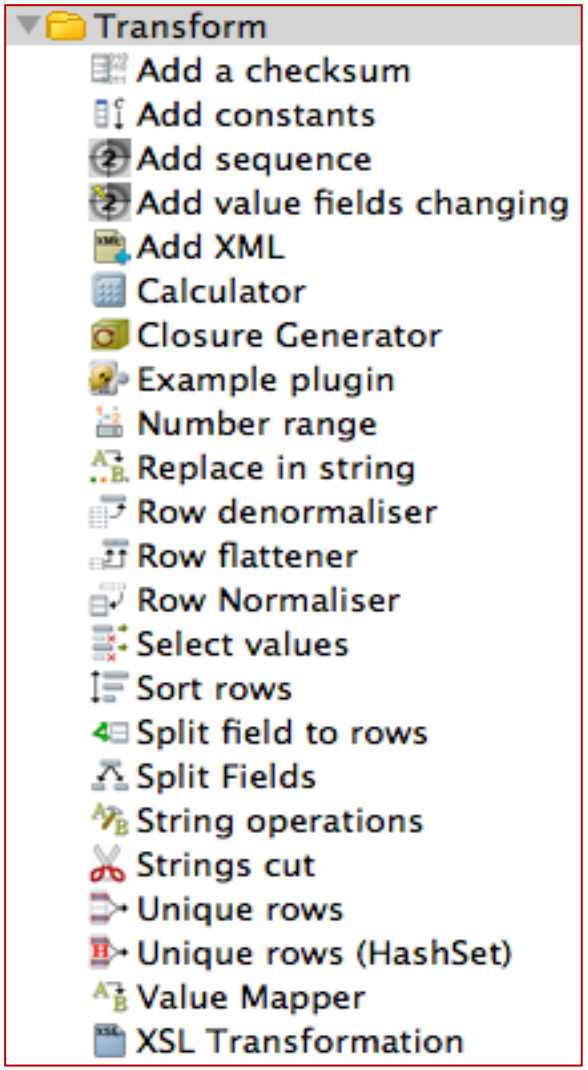
Kettle – Trasformazioni



Input

Raccolta di step che si occupano della ***gestione dei dati in ingresso***, presenti in varie tipologie.

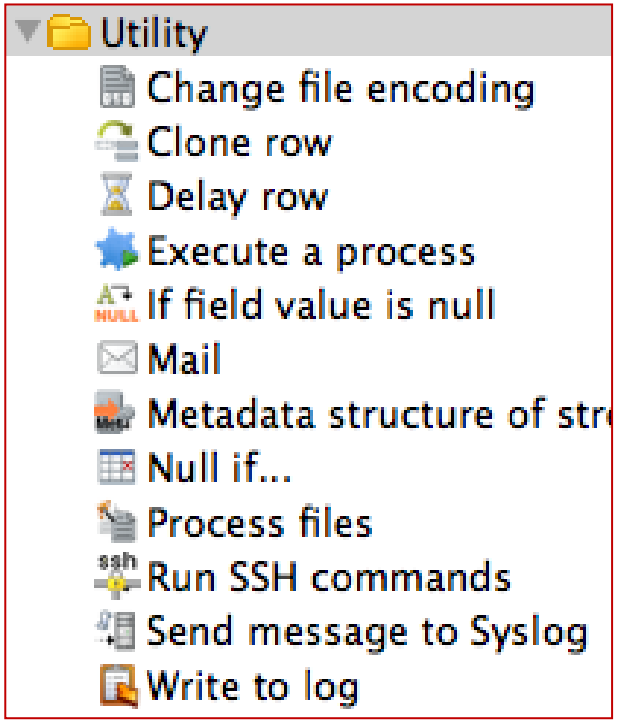
Kettle – Trasformazioni



Transform

insieme di step che si occupano di ***realizzare delle trasformazioni sui dati***, esempio separazione di campi, troncamento di stringhe, ordinamento di righe etc.

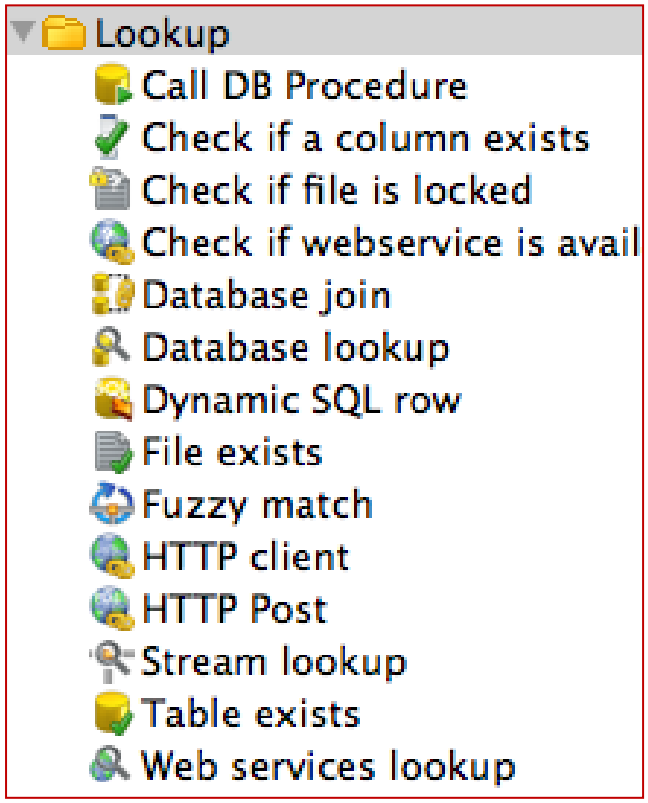
Kettle – Trasformazioni



Utility

insieme di step con ***funzionalità avanzante o di supporto***, esempio scrittura di log, controllo se un campo è null, cancellazione di righe etc.

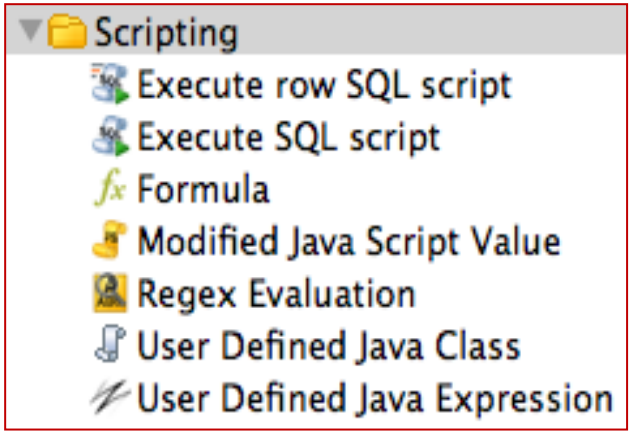
Kettle – Trasformazioni



Lookup

insieme di step che ***permettono delle operazioni di consultazione di dati*** su specifiche soluzioni o dati già estrapolati e mantenuti in strutture temporanee (aumentare velocità e reattività).

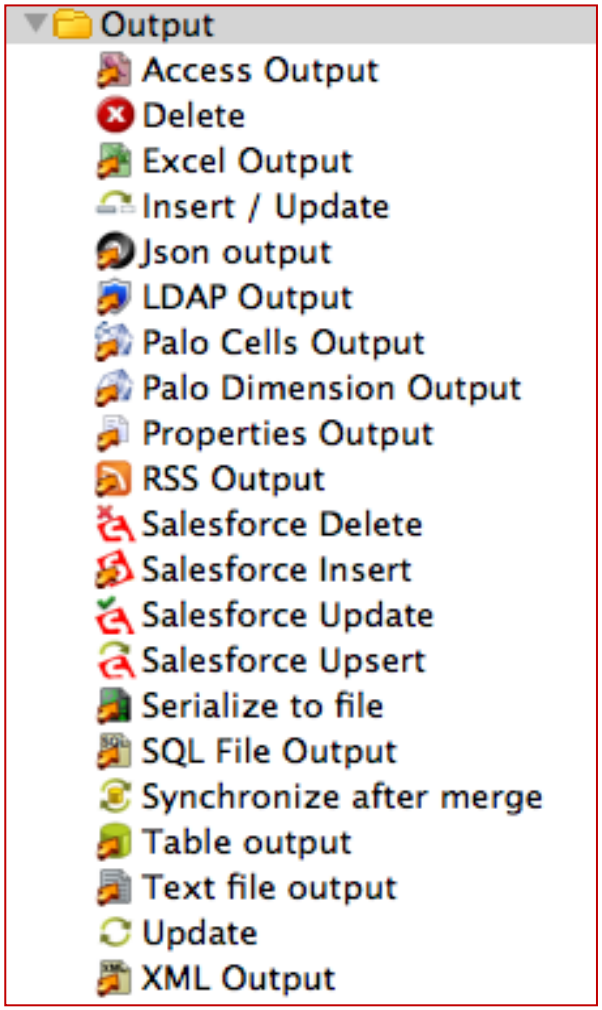
Kettle – Trasformazioni



Scripting

insieme di step in cui si ***possono definire degli script*** in diversi linguaggi.

Kettle – Trasformazioni



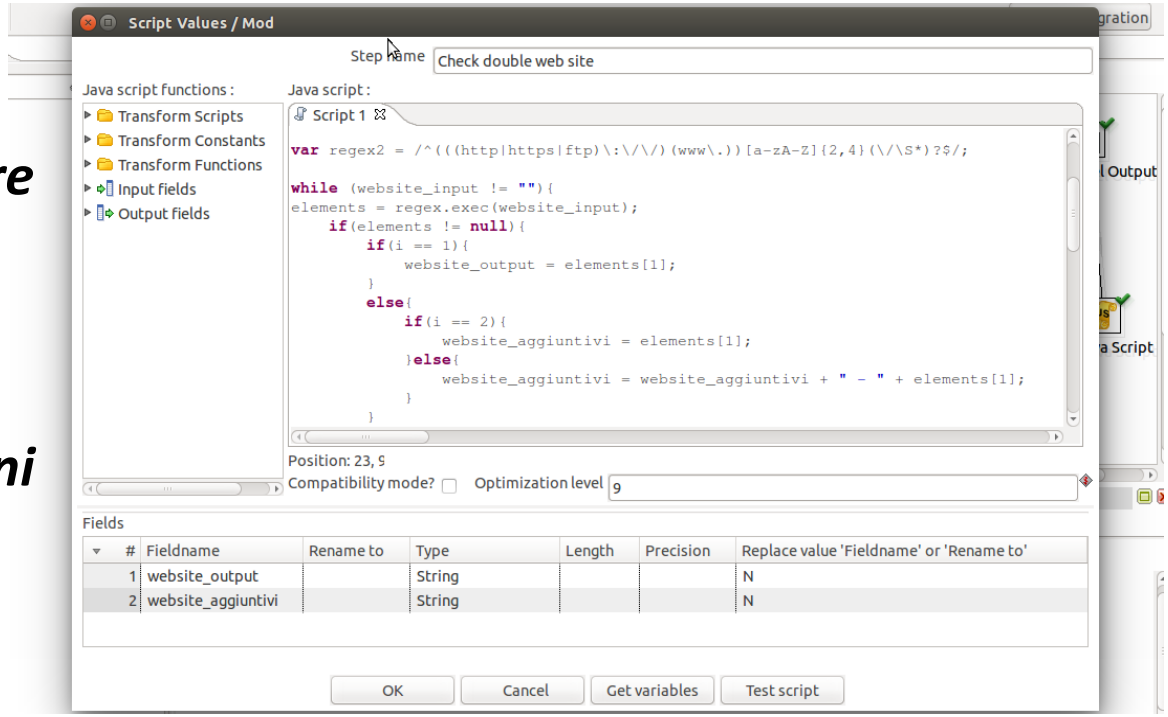
Output

Raccolta di step che si occupano della ***gestione dei dati in uscita***, presenti in differenti tipologie.

Kettle

Kettle mette a disposizione molte tipologie di step per le trasformazioni per varie operazioni sui dati, inoltre offre la possibilità:

- ***Utilizzare e aggiungere del codice Javascript.***
- ***Utilizzare le espressioni regolari.***



Kettle – Pan e Kitchen

- Le ***trasformazioni*** realizzate con ***Spoon***, possono essere lanciate con ***pan*** da riga di comando (analogamente si usa ***kitchen*** per i ***Job***).

```
/usr/local/pdi/pan.sh -file /home/pentaho/repos/LetturaDati.ktr
```

```
# Lancia il job ogni sabato alle sei di mattina...
6 6 * * 6 /usr/local/pdi/kitchen.sh -file /home/pentaho/repo/Aggiorna1.kjb >> /tmp/cron1.log 2>&1
```

- L'output viene tipicamente ridiretto su un log ed analizzato in caso di problemi.

```
INFO  07-06 06:10:02,486 - Using "/tmp/vfs_cache" as temporary files store.
INFO  07-06 06:10:02,712 - Pan - Start of run.
INFO  07-06 06:10:02,902 - Lettura dati per DWH - Dispatching started for transformation [Lettura dati per DWH]
INFO  07-06 06:10:02,929 - Lettura dati per DWH - This transformation can be replayed with replay date: 2011/06/07 06:10:02
INFO  07-06 06:10:03,233 - DB DWH - Connected to database [Self DB] (commit=100)
INFO  07-06 06:10:03,599 - DB AS_UTIL - Finished reading query, closing connection.
INFO  07-06 06:10:03,614 - DB AS_UTIL - Finished processing (I=27, O=0, R=0, W=27, U=0, E=0)
INFO  07-06 06:10:03,625 - DB DWH - Finished processing (I=0, O=27, R=27, W=27, U=0, E=0)
INFO  07-06 06:10:03,626 - Pan - Finished!
INFO  07-06 06:10:03,627 - Pan - Start=2011/06/07 06:10:02.713, Stop=2011/06/07 06:10:03.126
INFO  07-06 06:10:03,627 - Pan - Processing ended after 0 seconds.
INFO  07-06 06:10:03,627 - Lettura dati per DWH -
INFO  07-06 06:10:03,627 - Lettura dati per DWH - Step DB AS_UTIL.0 ended successfully, processed 27 lines. ( - lines/s)
INFO  07-06 06:10:03,628 - Lettura dati per DWH - Step DB DWH.0 ended successfully, processed 27 lines. ( - lines/s)
```

Kettle – Esercizio

- Creazione di una trasformazione che prende i dati da un file CSV e li memorizza in una tabella su Hbase.
- Creazione di un Job con all'interno una trasformazione di modifica.
  - Recupero dati da Hbase
  - Filtraggio su uno specifico campo
  - Correzione dei valori di un campo
  - Produzione di un file Excel di output
  - Memorizzazione dei dai corretti su HBase