A workflow example

In this example there will be 3 jobs, chained up in one work flow script, logically those 3 jobs need to execute in sequential order, the trick for chaining up sequential jobs would be introduced in this demonstration as well.

1. 1st job: import data into HDFS

using sqoop to import data from database to hdfs

```
cwlVersion: v1.0
class: CommandLineTool
baseCommand: sqoop-import
inputs:
  conect_flag:
    type: string
    inputBinding:
      position: 1
  connect_address:
    type: string
    inputBinding:
      position: 2
  sqoopMapper_flag:
    type: string
    inputBinding:
      position: 3
  sqoopMapper_number:
    type: int
    inputBinding:
      position: 4
  username_flag:
    type: string
    inputBinding:
      position: 5
  username_value:
    type: string
    inputBinding:
      position: 6
  password_flag:
    type: string
    inputBinding:
      position: 7
  password_value:
    type: string
    inputBinding:
      position: 8
  table_flag:
    type: string
    inputBinding:
      position: 9
  table_value:
    type: string
    inputBinding:
      position: 10
```

```
  targetdir_flag:
    type: string
    inputBinding:
      position: 11
  targetdir_value:
    type: string
    inputBinding:
      position: 12
requirements:
  - class: EnvVarRequirement
    envDef:
    - envName: JAVA_HOME
      envValue: "/usr/lib/jvm/java-8-openjdk-amd64"
    - envName: HADOOP_USER_NAME
      envValue: "hdfs"
    - envName: HBASE_HOME
      envValue: "/"
    - envName: HCAT_HOME
      envValue: "/"
    - envName: ACCUMULO_HOME
      envValue: "/"

outputs:
  execution_code:
```

```
      type: string[]

successCodes: [ 0 ]
```

2. 2nd job: execute map reduce on imported job in 1st step

map reduce by hadoop commands

```
cwlVersion: v1.0
class: CommandLineTool
baseCommand: hadoop
arguments: [jar,
"/usr/hdp/current/hadoop-mapreduce-client/hadoop-streaming.jar"]
inputs:
  input_flag:
    type: string
    inputBinding:
      position: 1
  input_path:
    type: string
    inputBinding:
      position: 2
  output_flag:
    type: string
    inputBinding:
      position: 3
  output_path:
    type: string
    inputBinding:
      position: 4
  mapper_flag:
    type: string
    inputBinding:
      position: 5
  mapper_script:
    type: string
    inputBinding:
      position: 6
  reducer_flag:
    type: string
    inputBinding:
      position: 7
  reducer_script:
    type: string
    inputBinding:
      position: 8
requirements:
  - class: EnvVarRequirement
    envDef:
    - envName: JAVA_HOME
      envValue: "/usr/lib/jvm/java-8-openjdk-amd64"
    - envName: HADOOP_USER_NAME
      envValue: "hdfs"

outputs:
  execution_code:
    type: string[]

successCodes: [ 0 ]
```

3. 3rd job: download map reduce result from hdfs to local file system

dowload file from hdfs by 'hadoop fs' command

```
cwlVersion: v1.0
class: CommandLineTool
baseCommand: hadoop
arguments: [fs, "-get"]
inputs:
  infile_path:
    type: string
    inputBinding:
      position: 1
  outfile_path:
    type: string
    inputBinding:
      position: 2
requirements:
  - class: EnvVarRequirement
    envDef:
    - envName: JAVA_HOME
      envValue: "/usr/lib/jvm/java-8-openjdk-amd64"
    - envName: HADOOP_USER_NAME
      envValue: "hdfs"

outputs:
  execution_code:
    type: string[]

successCodes: [ 0 ]
```

Notes for job type script

a. key word [class]: defines what type job it is
b. key word [baseCommand]: defines which command line the script will invoke
c. key word [arguments]: optional, if there is static arguments for the command, they can be defined as arguments.
d. key word [inputs]: defines input parameters, [position] defines the parameter position in whole command line, start from 1
e. key word [requirements], optional, here we use [EnvVarRequirement] feature to pass environment variables
f. key word [outputs] here we define a variable of [execution_code], it is not necessary for command type job, in fact the all outputs in given example are empty, however, it is useful in workflow script. The outputs from ancestor script is set as one of the input of descendant script, in such way, jobs will be executed sequentially, otherwise all scripts will be executed in parallel.
g. key word [successCodes]: defines success codes for command line, when command execution return code is beyond defined success codes, the script execution will be considered as fail.

4. workflow script to chain these 3 jobs up

workflow script

```
cwlVersion: v1.0
class: Workflow
inputs:
```

```
    conect_flag: string
    connect_address: string
    sqoopMapper_flag: string
    sqoopMapper_number: int
    username_flag: string
    username_value: string
    password_flag: string
    password_value: string
    table_flag: string
    table_value: string
    targetdir_flag: string
    targetdir_value: string
    input_flag: string
    input_path: string
    output_flag: string
    output_path: string
    mapper_flag: string
    mapper_script: string
    reducer_flag: string
    reducer_script: string
    infile_path: string
    outfile_path: string
  outputs: []

  steps:
    sqoopImport:
      run: sqoopImportJob.cwl
      in:
        conect_flag: conect_flag
        connect_address: connect_address
        sqoopMapper_flag: sqoopMapper_flag
        sqoopMapper_number: sqoopMapper_number
        username_flag: username_flag
        username_value: username_value
        password_flag: password_flag
        password_value: password_value
        table_flag: table_flag
        table_value: table_value
        targetdir_flag: targetdir_flag
        targetdir_value: targetdir_value
      out:  [execution_code]

    hadoop:
      run: mapReduceJob.cwl
      in:
        trigger: sqoopImport/execution_code
        input_flag: input_flag
        input_path: input_path
        output_flag: output_flag
        output_path: output_path
        mapper_flag: mapper_flag
        mapper_script: mapper_script
        reducer_flag: reducer_flag
```

```
      reducer_script: reducer_script
    out:  [execution_code]

  fetchResult:
    run: fetchResult.cwl
    in:
      trigger: hadoop/execution_code
```

```
      infile_path: infile_path
      outfile_path: outfile_path
    out:  []
```

a. key word [inputs]: defines all parameters are need in job scripts

b. key word [outputs]: defines final output for the whole flow

c. key word [steps]: defines sub job scripts, [run] defines corresponding script, [in] defines input parameters, [out] defines output for each step.

5. Input parameters and values

parameters

```
# sqoop job paramters
conect_flag: "--connect"
connect_address:
"jdbc:jtds:sqlserver:[redacted]1433/xa2be"
sqoopMapper_flag: "-m"
sqoopMapper_number: 1
username_flag: "--username"
username_value: "xa"
password_flag: "--password"
password_value: "xa"
table_flag: "--table"
table_value: "metadata_boolean"
targetdir_flag: "--target-dir"
targetdir_value: "/user/hdfs/metadata_boolean"

# map reducer job paramters
input_flag: "-input"
input_path: "/user/hdfs/metadata_boolean"
output_flag: "-output"
output_path: "/user/hdfs/findings"
mapper_flag: "-mapper"
mapper_script:
"/home/hui/test/cwl/mapReducer/tool_name_test_new_4_mapper.py"
reducer_flag: "-reducer"
reducer_script:
"/home/hui/test/cwl/mapReducer/tool_name_test_new_4_reducer.py"

# fetch finding from hdfs job parameters
infile_path: "/user/hdfs/findings/part-00000"
outfile_path: "/home/hui/test/cwl/mapReducerFlow/out"
```

parameter names are distinct