sqoop

`sqoop`

SQOOP安装与配置

下载sqoop 1.4.6

download and documentation .

配置

. 在`etc/profile`中增加环境变量 SQOOP_HOME

. 执行`cd /${SQOOP_HOME}/conf;cp sqoop-env-template.sh sqoop-env.sh`

. 修改`${SQOOP_HOME}/conf/sqoop-env.sh`,为其中的`HADOOP_COMMON_HOME`

`HADOOP_MAPRED_HOME` `HBASE_HOME` `HIVE_HOME`赋正确的值

PostgreSQL JDBC 驱动

由于使用的Java版本为***7u25***，故使用`JDBC41`这个版本的驱动： download

下载zookeeper 3.3.6

链接： download and documentation

配置:可以使用默认值

`cd zookeeper-3.3.6/conf; cp zoo_sample.cfg zoo.cfg`

启动zookeeper: `/root/t01965/zookeeper-3.3.6/bin/zkServer.sh start`

SQOOP IMPORT的使用

目的

import工具从RDBMS中导入一个单独的表至HDFS，表中每一行作为HDFS中一条单独的记录，每条记录可以存储为文本文件(每行一条记录)，或者是Avro、SequenceFile这样的二进制文件

语法 & 示例

```
sqoop import (generic-args) (import-args)
sqoop-import (generic-args) (import-args)
#----
sqoop import --connect jdbc:postgresql://207.101.67.46:5432/imos --table=tbl_vehicle_record_2016_06_08 --username postgres -password admin_123
```

在使用sqoop向HDFS中导入数据时会在当前目录生成一个与表同名的`.java`文件，这个文件可以用来解释sqoop导入生成的文本文件。

通用参数

```
--connect <jdbc-uri> #JDBC
--connection-manager <class-name> #Class
--driver <class-name> #Driver
--hadoop-mapred-home <dir> #$HADOOP_MAPRED_HOME
--help
--password-file #指定包含验证密码的文件
-P #从控制台输入密码
--password <password>
--username <username>
--verbose #任务处理时打印更多细节信息
--connection-param-file <filename> #选项属性文件
--relaxed-isolation #事务隔离
--password-file ${user.home}/.password
--options-file <filename>
```

导入控制参数

```
--append #数据追加至HDFS中已存在的数据集中
--as-avrodatafile #保存为Avro数据文件
--as-sequencefile #保存为序列化文件
--as-textfile #保存为文本文件(默认 )
--as-parquetfile #保存为parquet文件
--boundary-query <statement> #
--columns <col, col, col> #指定要导入的列
--delete-target-dir #目标目录若存在则先删除
--direct
--fetch-size <n> #一次从数据库读取的条目数
--inline-lob-limit <n> #设置内联LOB最大值
-m,--num-mappers <n> #使用n个mapper并行执行
-e,--query <statement> #导入结果
```

```
--split-by <column-name> #用来分隔任务单元的列名
--autoreset-to-one-mapper #若表无主键并且未指定--split-by，则使用一个mapper执行
--table <table-name> #要导入的表名
--target-dir <dir> #HDFS中的目标目录
--warehouse-dir <dir> #HDFS中目标目录的你目录
--where <where clause> #导入过程中使用的WHERE子句
-z,--compress #压缩使能
--compression-codec <c> #使用Hadoop自带的压缩解压工具(默认gzip)
--null-string <null-string> #空值写入的字符串,默认为""
--null-non-string <null-string> #(默认为"")
```

密码保护

Hadoop 2.6.0后提供API给密码加密并生成加密文件(2.2.0版本好像没有？)

数据库配置参数

使用JDBC连接数据库时，可以使用`--connection-param-file`指定额外的参数，文件内容是标准的Java属性文件，参数在建立连接时被加载

选择要导入的数据

- 通过`--table`指定要导入的表，默认情况下所有的列都会被导入，导入的结果是按行逗号分隔的
- 可以用使用`--columns "colName1,colName2,colName3"`这种方式指定要导入的列(导入结果中列的顺序是按照`--columns`指定顺序)，
  可以使用`--where "condition"`选择要导入的行，示例：

```
sqoop import --connect jdbc:postgresql://207.101.67.46:5432/imos --table=tbl_vehicle_record_2016_06_08 --where "record_id < 500" --username postgres -password admin_123 --target-dir=/tbl_vehicle_record_2016_06_08
```

- 使用`--query`选项，可以导入任意一条SQL语句的结果，示例：

```
sqoop import \
  --query 'SELECT a.*, b.* FROM a JOIN b on (a.id == b.id) WHERE $CONDITIONS' \
  --split-by a.id --target-dir /user/foo/joinresults
  #注意：若使用双引号则应在'$'前加'\'以防止shell认为$CONDITIONS是个shell变量
```

并行控制

sqoop从其它数据库导入数据时是并行的，可以通过`-m`或`--num-mappers`来指定map任务的个数，默认为4个。
在执行并行导入时，sqoop需要一个拆分工作量的标准，sqoop使用一个'拆分列'来拆分工作量(默认为主键这列)，sqoop会获取它的最大和最小值，然后把任务均分为`-m`份。若主键实际值不均匀分布会引起任务分配不均，这时需要通过`--split-by`显式指定一个(均匀分布的)列。若表没有主键并且没有指定`--split-by`，则只能指
定`-m 1`或`--autoreset-to-one-mapper`,否则任务无法执行。

导入过程控制

默认使用JDBC执行导入，某些数据库可以使用数据库指定的工具执行导入以提供更高性能(如mysql的mysqldump)，具体细节可以查询相关数据库文档。
sqoop默认在导入表时会在HDFS的用户主目录下建立同名文件夹，相关文件放在此目录下。可以通过`--warehouse-dir`指定父目录。
`--target-dir`可以指定一个与table不同名的目录
`--default-character-set`可以指定字符集。
若HDFS中存在目的文件夹，则导入会失败，可以重新指定目录或使用`--append`以追加方式导入。

类型映射

`--map-column-java` & `--map-column-hive`
语法：`--map-column-java <name of column>=<new type>`
例如：`sqoop import ... --map-column-java id=String,value=Integer`

增量导入

sqoop有增量导入模式，可以用来导入那些比前导入的更新的行。
参数：

```
1.  --check-column (col)     #Specifies the column to be examined when dete
    rmining which rows to import. (the column should not be of type CHAR/N
    CHAR/VARCHAR/VARNCHAR/ LONGVARCHAR/LONGNVARCHAR)
2.  --incremental (mode)    #Specifies how Sqoop determines which rows are
    new. Legal values for mode include append and lastmodified.
```

```
--last-value (value) #Specifies the maximum value of the check column from the previous import.
```

文件格式

文本文件导出的分隔符可以是逗号(默认)，TAB或其它字符，如下

```
Argument     Description
--enclosed-by <char>     Sets a required field enclosing character
--escaped-by <char>  Sets the escape character
--fields-terminated-by <char>    Sets the field separator character
--lines-terminated-by <char>     Sets the end-of-line character
--mysql-delimiters   Uses MySQL's default delimiter set: fields: , lines: \n escaped-by: \ optionally-enclosed-by: '
--optionally-enclosed-by <char>  Sets a field enclosing character
```

大型对象处理

sqoop 对大型对象的处理比较特殊，如果数据很大，则它们不应该在内存中实例化，而应该流式处理。大型对象可以与其他的数据存储在一起，在这种情况下，它们在每个访问中都会被完全地在内存中实例化，也可以被存储在与主数据存储相关的二次存储文件中。默认小于16M的数据与其它数据存储在一起，再大一点的数据存储在目标目录中以_lobs结尾的子目录中，这些文件以单独的经过优化的格式存储，可以大至2^63Bytes.

其它导入工具

- sqoop-import-all-tables:此工具用来一次性导入多个表，使用的前提条件：
  1. 每张表都必须有主键
  2. 所有的列都要被导入
  3. 不能使用非默认的分隔列(主键)，也不能通过`WHERE`添加附加条件。
- sqoop-import-mainframe: 此工具用来从大型(网络)主机中导入所有被分隔的连续的数据集，源数据必须是文本文件。

其它工具

```
sqoop-codegen
sqoop-create-hive-table
```

```
sqoop-eval
sqoop-export
sqoop-help
sqoop-job
sqoop-list-databases
sqoop-list-tables
sqoop-merge #把两个数据集整合成一个
sqoop-metastore #用来配置Sqoop主机共享的元数据存储库
sqoop-version
```