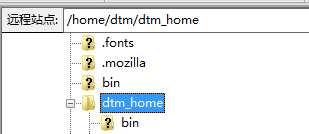
**据传输任务总结**

一．传输任务的创建修改执行流程

传输任务执行中的常量解释：

${transfertype} ： 传输任务类型小写

${%DTM\_HOME%} ： dtm环境根目录

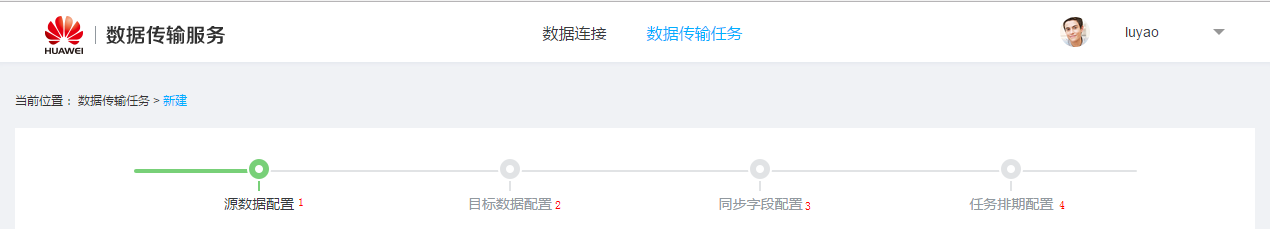


${传输任务job id} ：传输任务执行实例的id



1. 数据传输任务的创建
   1. 传输任务创建页面简介和意见

创建一个数据传输任务需要配置4部分



1. 源数据配置

支持hive/ftp/oracle/local/email五中数据链接

1. 目标数据配置

支持ftp/hbase/oracle/hive四中数据链接

1. 同步字段配置

配置从源数据到目标数据的映射关系

1. 任务排期配置

配置任务排期

个人意见：传输任务创建页面的的个步骤是我觉得dtp模块最差的地方。

1. 创建一个传输任务的时候，是通过第一步和第二部的排列组合，来确定你准备创建的传输任务格式，如果你的排列组合不对，在步骤1，2中选择好数据链接的时候才会提示不支持该类型的传输任务如下图所示。配置数据链接和目标数据链接的时候会把所有的链接都展示出来然后用户自己去选择链接，如果用户创建链接的时候没有特别标示，选择链接的时候都不知道对应链接的类型。



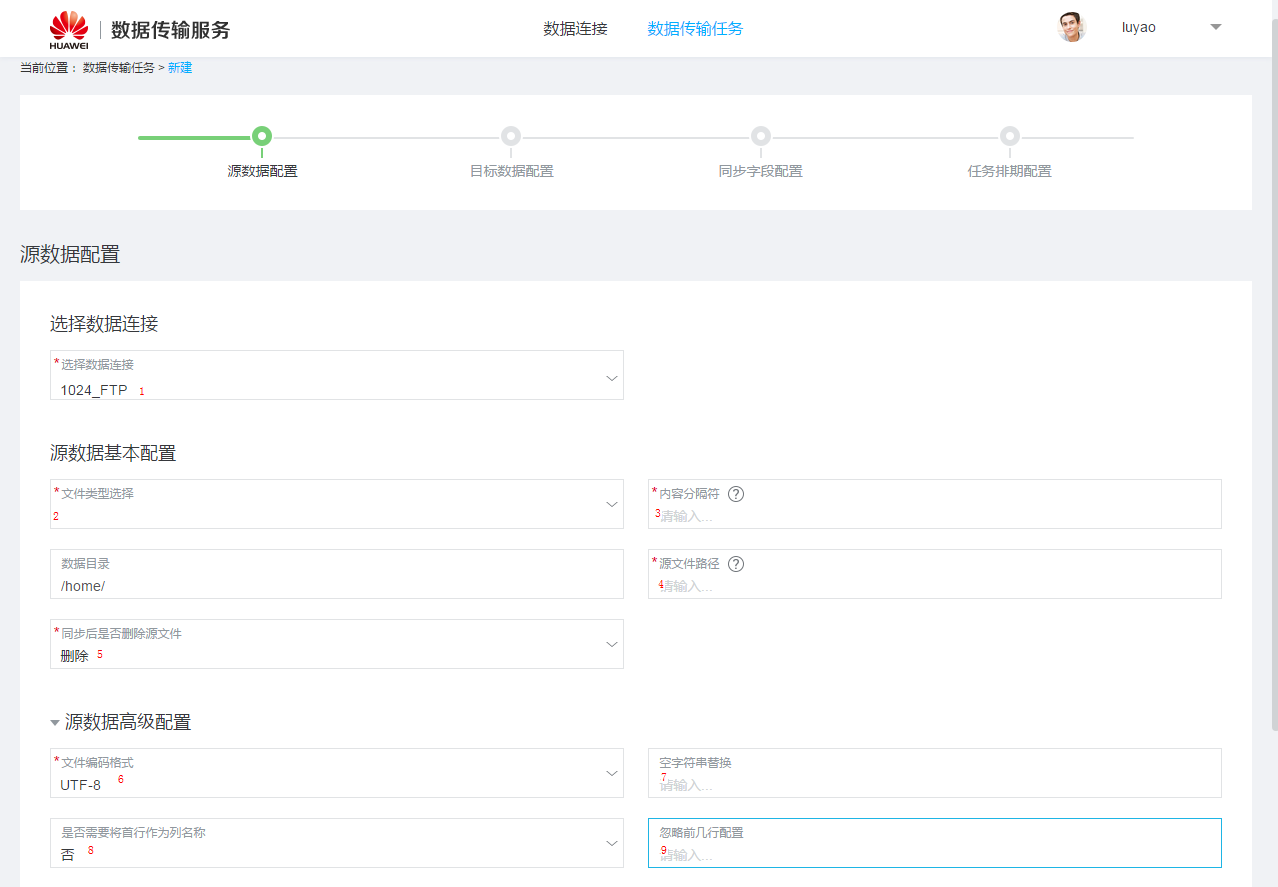
个人看来一个传输任务的创建应该由5个步骤构成更为合适

传输任务类型-》源数据配置-》目标数据配置-》同步字段配置-》配置任务排期

第一步传输任务类型选择连确定你要创建的传输任务类型，之后源数据配置和目标数据配置中的数据链接时只能展示你选择的对应类型的数据链接。

1. 所有的传输任务前台全部写在一个html和js中，html写了2000行，js有5000行，修改起来十分困难，后期传输任务维护，新增一种类型需要把每种类型的传输任务全都修改一边，很麻烦。
   1. 各类型传输任务的创建

* FtpToHive
* 源数据配置：



1 .选择指定的ftp链接

2.选择文件类型，支持txt/tsv/csv(文本类型格式)，gz/tar.gz(压缩文件)；压缩文件解压后请保证为txt/tsv/csv格式。

3.内容分隔符，txt类型文件的需要自己配置，tsv文件默认为“\t”，csv文件默认为“，”。

4.源文件路径支持配置多个目录，目录和文件名支持通过时间字段动态匹配。具体配置方式点击原文件路径后面的“？”有展示。

关于时间字段动态匹配，正常文件路径和文件名中是不支持“<”和“>”符号的，这里用该符号框住需要动态替换的时间字段。如果填写错误，前台会出现校验异常的提示。

时间字段的动态匹配主要为了一些周期性任务的处理，通过文件名称或者路径的变更，保证获取最新的数据。

5.同步后是否删除文件，支持删除/备份/不删除三种设置，如果选择备份，则需要填写备份路径，备份路径的个数应该和源文件路径的个数一致。

6.文件编码格式，支持UTF-8/GBK，默认为UTF-8。

7-9 ftptohive传输任务暂时不支持该配置。

* 目标数据配置：



1. 选择指定的hive链接，
2. 选择好hive链接以后，选择指定的hive链接数据库
3. 选择对应的hive表
4. 选择加载方式，支持追加（在原有表中继续增加数据）/覆盖（删除原有表中的数据）

* 同步字段配置：



1. 输入字段，该字段没有实际意义，主要用于和字段的序号相互匹配，后续表达式计算中匹配的是输入字段对应序号的字段值，如果文件中有10个字段，传输任务使用前8个，其中第6个字段不需要输出，那么序号1-8的输入字段都得填一个输入字段值。
2. 输出字段，指对应的输入字段需要保存到hive表中的哪个字段中。
3. 计算规则，支持原字段输出/表达式计算/常量

如果是原字段输出，按照字段将ftp文件中对应的字段映射到hive表的对应字段中

如果是表达式计算，表达式中填写的计算值为对应的输入字段，可以是本行的输入字段也可以是之前行的输入字段。

如果是常量输出，则将对应

1. 规则表达式，对应3中的规则

如果源字段输出，则默认为输入字段

如果是表达式计算，可以选择对应的表达式配合输入字段进行计算

如果是常量，这里需要填写对应的常量值。

1. 是否输出，ftptohive的传输任务需要按照顺序使用ftp文件中的字段，如果ftp文件中有5个字段，其中第三个字段不需要使用，则字段序号3的是否输出选择否即可。
2. 设为分区，如果hive表为分区表，选择否的时候，输出字段下拉框只显示hive表中非分区字段，选择是的时候输出字段下拉框中只显示hive表中的分区字段。

其他：

配置的时候可能还存在一些细节问题，这里不太会用文字描述，可以直接询问李浩

关于表达式计算中的表达式选择可以询问李云志，他比较清楚。

* 任务排期配置：



1. 执行策略：任务执行方式

支持周期性执行：年/月/周/天/小时/自定义

执行一次：立即执行/执行一次

工作流触发：由dam测的工作流任务进行调度执行。

1. 执行时间

配置对应策略的执行时间，其中自定义的时候如果配置“\* \* \* \* \* \*”，则表示该任务每分钟执行一次。

1. 有效期开始时间，任务在有效期外不会执行。
2. 有效期结束时间，任务在有效期外不会执行。
3. 任务名称，全局唯一，请勿和其他任务重名。
4. 描述，可有可无。

* 任务涉及表：

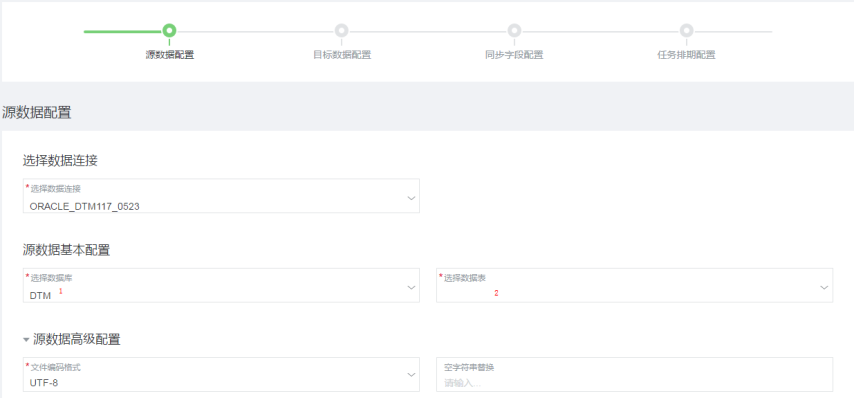
T\_DATATRANSFERS

T\_DATATRA\_SRCFTP

T\_DATATRA\_DSTHIVE

T\_DATATRA\_DATATRAMAP

* OracleToHive
* 源数据配置：



* + 1. 选择数据库
    2. 选择表

注意：这里的oracle数据库，和hive/hbase数据库不一样，在dam测没有创建入口，需要从后台进行创建。

* 目标数据配置：

见ftptohive的目标字段配置。

* 同步字段配置：



* + - 1. 输入字段为oracle表中的字段
      2. 输出字段为对应的hive表中的字段
      3. 输入字段类型通过输入字段自动映射
      4. 输出字段类型通过输出字段自动映射

注意，这里需要关注输入字段的类型可以映射到输出字段中，比如输入类型是varchar2，输出类型为number，任务很可能执行失败。

* + - 1. 计算表达式，支持原字段/常量/表达式计算
      2. 规则表达式，对应计算规则，原子段则自动匹配输入字段，表达式则可以使用指定的表达式对输入字段进行处理，常量则需要自己填写常量的默认值
      3. 是否输出，这里如果不输出建议直接删除该列，该任务和ftptohive的传输任务不一样，从文件中读取数据，需要根据序号一个个获取字段，否则字段会不对，数据库则没有该问题，想取哪个字段就取哪个字段，不输出的直接不列出来即可。
      4. 是否设为分区，选否时输出字段下拉框中的为hive表中非分区字段，选是是输出字段下拉框中为hive表中的分区字段。
      5. 允许为空，该配置暂时没有意义。
* 任务排期配置：

见ftptohive的任务排期配置。

* 任务涉及表：

T\_DATATRANSFERS

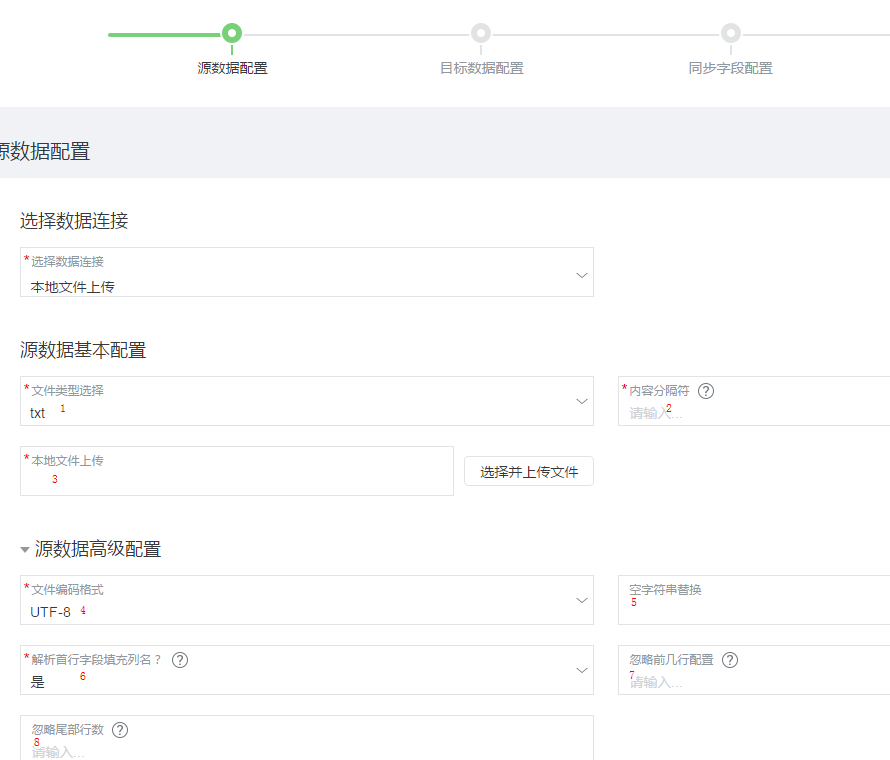
T\_DATATRA\_SRCORACLE

T\_DATATRA\_DSTHIVE

T\_DATATRA\_DATATRAMAP

T\_DT\_SORACLE\_PARTITIONINFO（oracle表存在分区筛选时使用）

* LocalToHive
* 源数据配置：



* + - * 1. 文件类型，支持txt/csv/tsv/xls
        2. 内容分隔符，文件类型为txt格式时可配，其他格式文件有默认值
        3. 选择本地文件上传，本地文件不可过大，最多100M，大文件需要使用ftptohive传输任务

文件上传以后就会进行一次解析，上传文件之前需要优先配置内容分隔符。

* + - * 1. 文件编码格式，支持UTF-8和GBK，由于文件需要预处理，实际到MR任务时候的文件预处理后会转换成UTF-8格式。
        2. 空字符串替换，暂时没用
        3. 解析首行字段填充列名：选择是的话，文件里面第一行会自动加载到同步字段配置中。
        4. 忽略前几行配置：文件预处理后会忽略开头的几行
        5. 忽略尾部行数：文件预处理后会忽略结尾的几行配置
* 目标数据配置：

见ftptohive的目标字段配置。

* 同步字段配置：

见ftptohive的同步字段配置。

* 任务排期配置：

见ftptohive的任务排期配置。

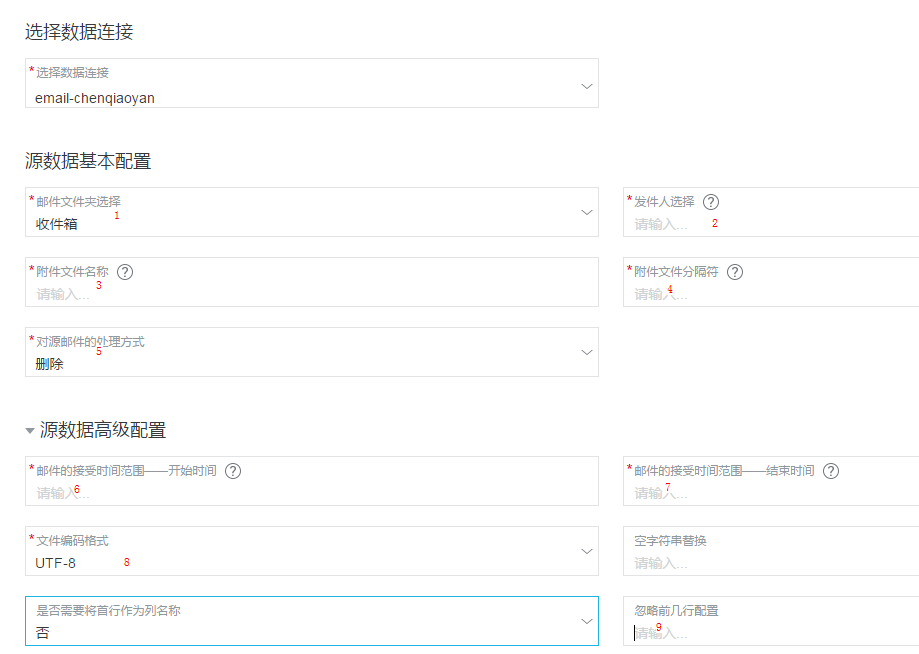
* 任务涉及表：

T\_DATATRANSFERS

T\_DATATRA\_SRCLOCAL

T\_DATATRA\_DSTHIVE

T\_DATATRA\_DATATRAMAP

* EmailToOracle
* 源数据配：

1. 选择邮箱文件夹
2. 选择发件人
3. 选择附件文件名称：文件支持txt/csv/tsv/xls格式，文件中支持通配符
4. txt格式文件需要自己输入分割符
5. 文件支持删除，备份，标记已读和不做处理
6. 接收邮件开始时间，支持根据时间动态匹配，动态匹配的时间是传输任务执行时间

时间格式为 年月日时分秒

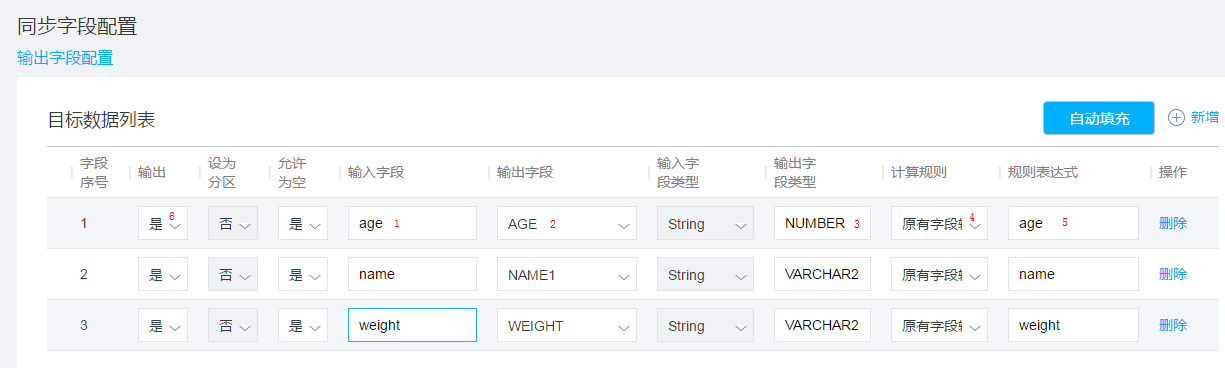
1. 接收邮件结束时间，支持根据时间动态匹配，动态匹配的时间是传输任务执行时间
2. 文件编码格式，支持UTF-8/GBK，对于GBK格式的文件存在一个预处理操作，实际上传到hdfs上进行MapReduce任务的文件全是UTF-8格式的编码
3. 忽略前几行配置

* 目标数据配置：



1. Oracle链接
2. 选择数据库
3. 选择表
4. 数据加载方式支持覆盖/追加(注意，如果任务失败，有个数据回滚机制)

* 同步字段配置：



1. 输入字段，可以填写任意值，作为一个标示和字段序号对应
2. 输出字段，oracle表中对应的字段
3. 输出字段类型，和输出字段自动映射
4. 计算规则，支持原子段输出/表达式计算/常量
5. 规则表达式，对应计算规则，原子段则自动匹配输入字段，表达式则可以使用指定的表达式对输入字段进行处理，常量则需要自己填写常量的默认值
6. 是否输出，文件中的字段是根据顺序输出的，如果中间的某个字段不使用，请在这里选否。

* 任务排期配置：

见ftptohive的任务排期配置。

* 任务涉及表：

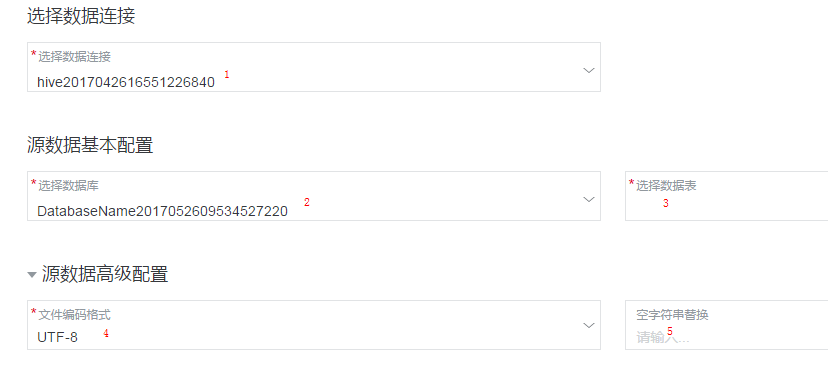
T\_DATATRANSFERS

T\_DATATRA\_SRCEMAIL

T\_DATATRA\_DSTORACLE

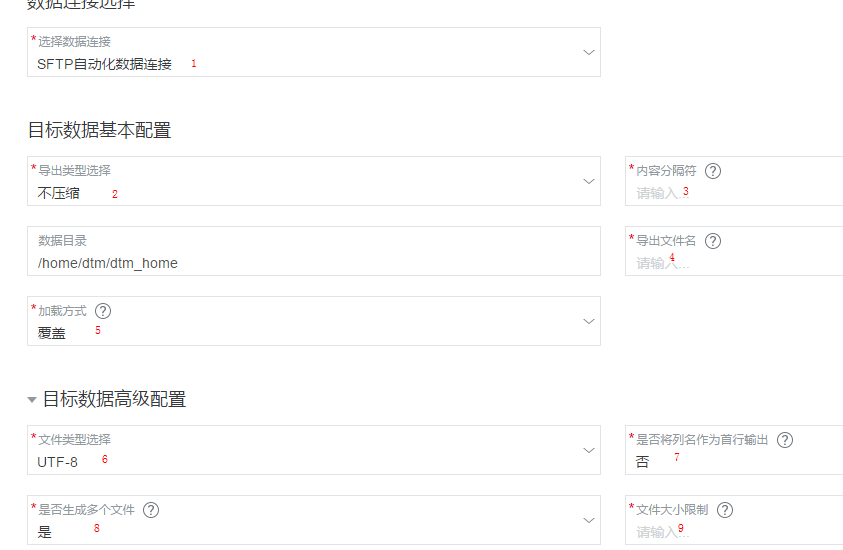
T\_DATATRA\_DATATRAMAP

* HiveToFtp
* 源数据配置：

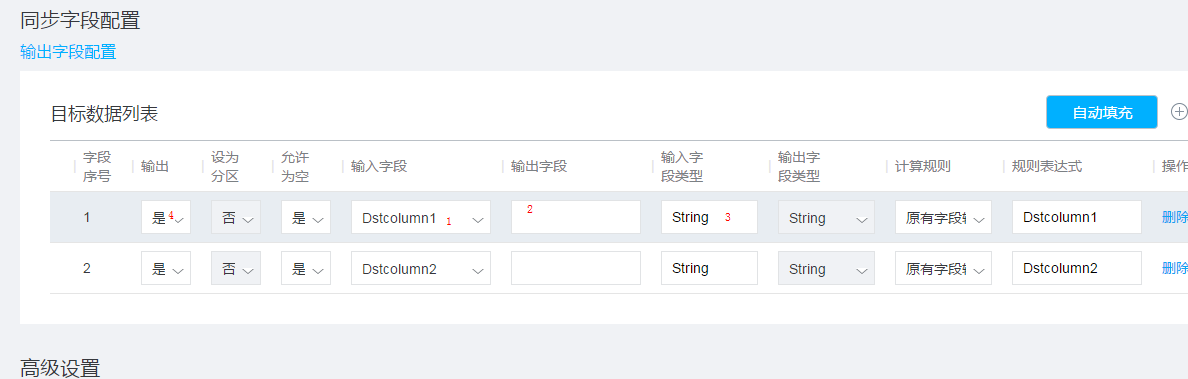


1. 选择hive数据连接
2. 选择hive数据库
3. 选择hive数据库表
4. 存入hive数据库的数据默认编码格式是UTF-8，查出来的也只有UTF-8
5. 该功能不支持

* 目标数据配置：



* + - 1. 链接类型，选择ftp格式
      2. 是否压缩，不压缩或者压缩成gz格式
      3. 内容分隔符，自定义。
      4. 导出文件名，输出文件默认是文本格式，文件名前面可以补充文件路径，路径支持通过时间字段动态匹配，匹配的格式参照问号中的内容
      5. 加载方式只支持覆盖，会删除目录中同名文件
      6. 文件编码格式只支持UTF-8
      7. 是否将列名作为首行输出，选是列名在每个文件的首行输出
      8. 选择生成多个文件，则可以分文件输出，每个文件以-n结尾，从0递增
      9. 生成多个文件时文件的大小，可以是KB/MB/GB
* 同步字段配置：



1. 输入字段，hive表的输入字段
2. 输出字段，作为输出字段的标示，如果目标数据配置选择将列名作为首行输出，则这里的值将输出到文件首行
3. 输入字段类型和，根据输入字段自动匹配
4. 是否输出，选择否，则该字段不做输出，使用场景：表达式计算的时候选择两个以上的参数的时候，这里就可以选择不输出。

* 任务排期配置：

见ftptohive的任务排期配置。

* 任务涉及表：

T\_DATATRANSFERS

T\_DATATRA\_SRCHIVE

T\_DATATRA\_DSTFTP

T\_DATATRA\_DATATRAMAP

T\_DT\_SHIVE\_PARTITIONINFO（hive表存在分区筛选时使用）

* HiveToOracle
* 源数据配置：

见hivetoftp数据源配置

* 目标数据配置：

见EmailToOracle的目标数据配置

* 同步字段配置：



1. 输入字段，hive表中输入字段
2. 输出字段，输出到oracle表中的字段
3. 输入字段类型通过输入字段自动映射
4. 输出字段类型通过输出字段自动映射
5. 是否输出，选否可以不输出，仅作为表达式计算的一个参数

* 任务排期配置：

见ftptohive的任务排期配置。

* 任务涉及表：

T\_DATATRANSFERS

T\_DATATRA\_SRCHIVE

T\_DATATRA\_DSTORACLE

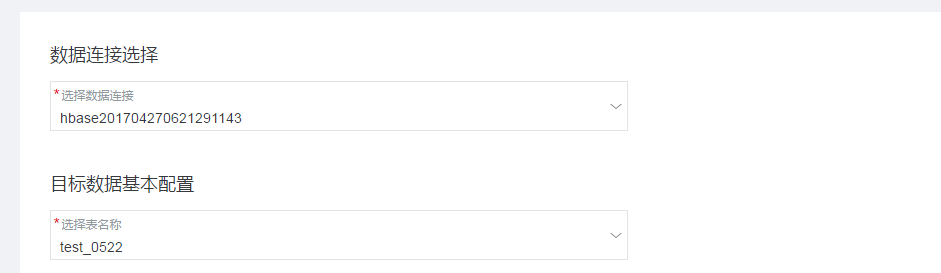
T\_DATATRA\_DATATRAMAP

T\_DT\_SHIVE\_PARTITIONINFO（hive表存在分区筛选时使用）

* HiveToHbase
* 源数据配置：

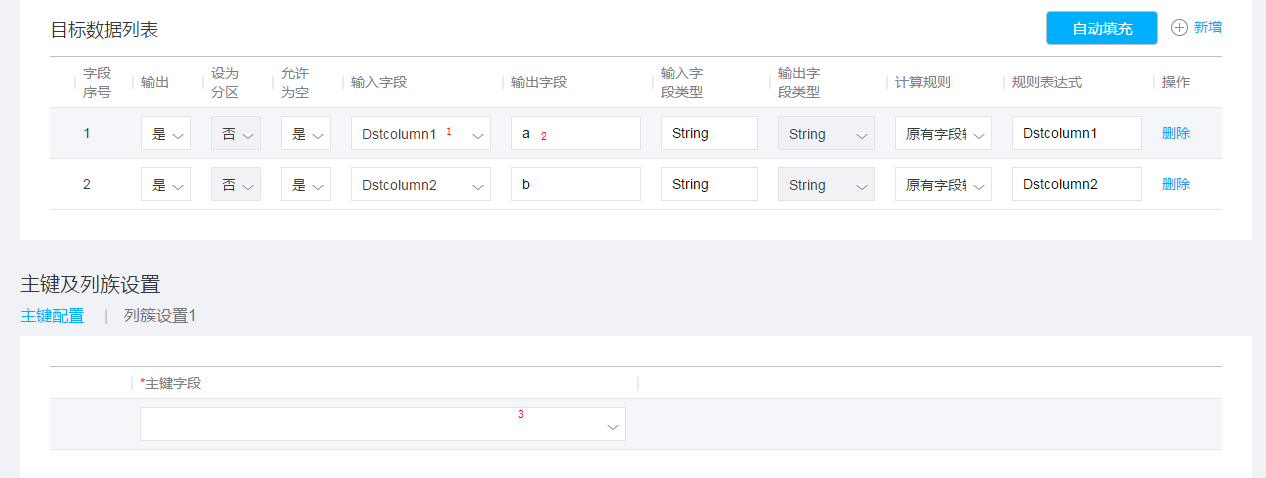
见hivetoftp数据源配置

* 目标数据配置：



1. 选择hbase数据连接
2. 选择对应的hbase表

* 同步字段配置：





1. hive表的输入字段
2. 输出到hbase表中的字段表示，请和hbase表中对应的列簇字段一致
3. 主键字段，下拉框中的值为输出字段的值
4. Hbase表的列簇，一个列簇中可以有多个字段
5. 列簇中的字段，下拉框中数据来源为2中输出字段
6. 列名和输出字段一致，如果hbase表中不存在，会自动创建。

* 任务排期配置：

见ftptohive的任务排期配置。

* 任务涉及表：

T\_DATATRANSFERS

T\_DATATRA\_SRCHIVE

T\_DATATRA\_DSTHBASE

T\_DATATRA\_DATATRAMAP

T\_DT\_SHIVE\_PARTITIONINFO（hive表存在分区筛选时使用）

1. 数据传输任务的修改

数据传输任务的创建流程和数据传输任务的修改流程基本一致。

修改传输任务下几个方面特点。

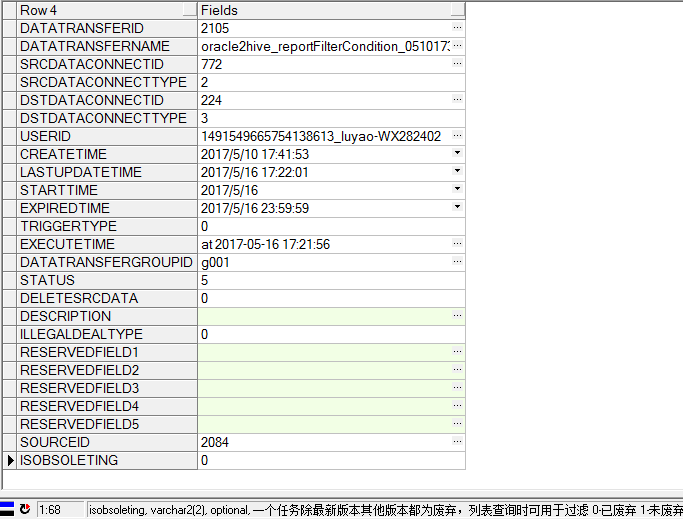
1只能修改自己的传输任务。

2修改的时候会把之前的传输任务配置全部展现在传输任务的几个配置页面上。

3修改的时候可以修改传输任务类型，相当于重新创建一个传输任务。

4localtohive的传输任务不支持修改，因为local文件需要即时上传有效，需要上传时候读取文件获取文件头信息。

5传输任务修改后，之前的传输任务信息依旧在t\_datatransfers中保存，之前的传输任务状态为已经废弃。



6新建的传输任务t\_datatransfers表中datatransferid = sourceid，修改后的不一样，通过sourceid可以知道之前的任务，用于查看修改过的传输任务之前的历史执行日志信息。

1. 传输任务的执行

* HiveToFtp

1. 从指定hive表中查询对应的数据，处理方式为在java测拼接指定的hql语句，链接到hive后执行hql将数据输出到指定hdfs文件夹。

Hsql为：INSERT OVERWRITE DIRECTORY '/transferInfo/${transfertype}/input/${传输任务job id}' select if(test1 is null,'', test1) test1 from ${数据库名}.${表名}

输出文件夹为：/transferInfo/${transfertype}/input/${传输任务job id}

1. 通过mapreduce任务，对输出参数进行相应的函数转换，将转换结果存放到hbase指定文件夹

Job 的名称为hivetoftp\_${传输任务job id}

输出文件夹为：/transferInfo/${transfertype}/output/${传输任务job id}

1. 从输出文件夹中读取文件数据，并写入本地文件中

本地文件临时存放位置：${%DTM\_HOME%}/hive\_temp\_path/${传输任务job id}

生成的文件支持按照时间动态替换文件名称，支持按照指定大小分文件

1. 将本地文件上传到ftp，ftp目录支持按照时间动态替换目录路径，文件上传完成后，会删除本地文件临时存放目录。

支持将文件压缩成gz格式再上传。

* HiveToHbase

1. 从指定hive表中查询对应的数据，处理方式为在java测拼接指定的hql语句，链接到hive后执行hql将数据输出到指定hbase文件夹。

Hsql为：INSERT OVERWRITE DIRECTORY '/transferInfo/${transfertype}/input/${传输任务job id}' select if(test1 is null,'', test1) test1 from ${数据库名}.${表名}

输出文件夹为：/transferInfo/${transfertype}/ftp/input/${传输任务job id}

1. 通过mapreduce任务，对输出参数进行相应的函数转换，将转换结果存放到hbase指定文件夹

Job 的名称为hivetohbase\_${传输任务job id}

输出文件夹为：/transferInfo/${transfertype}/hbaseout${传输任务job id}

1. 将文件存入指定文件由输出目录导入到指定hbase数据库中。

注意点：1）hbse数据库是在dam模块进行创建的

2）dtm模块建立的hbase链接是公用的，hbase链接时展示的是当前用户在dam处创建的hbase表

3）hbase数据库现在存在新老两种形式

比如一个用户名为1479976440194646157\_luyao-WX282402

该用户在V200R001C28版本前创建的hbase表aaa实际为

default:WX282402\_aaa

之后创建的表aaa实际为

1479976440194646157:aaa

* HiveToOracle

1. 从指定hive表中查询对应的数据，处理方式为在java测拼接指定的hql语句，链接到hive后执行hql将数据输出到指定hbase文件夹。

Hsql为：INSERT OVERWRITE DIRECTORY '/transferInfo/hivetooracle/input/${传输任务job id}' select if(test1 is null,'', test1) test1 from ${数据库名}.${表名}

输出文件夹为：/transferInfo/${transfertype}/input/${传输任务job id}

1. 通过mapreduce任务，对输出参数进行相应的函数转换，将转换结果存放到hbase指定文件夹

Job 的名称为hivetooracle\_${传输任务job id}

输出文件夹为：/transferInfo/${transfertype}/output/${传输任务job id}

1. 读取输出结果到本地，链接指定oracle数据库，将数据数据结果保存到oracle数据库中，数据存入oracle数据库之前需要进行对每个字段依据oracle表的字段进行校验，校验不通过的数据舍弃，使用的是batchupdate，不支持事务操作。

由于需要对每个字段进行校验是否能够正常存入数据库，效率较低

注意：当一个传输任务实例使用了一个oracle链接以后，其他的传输任务是不能使用该oracle表的，这里是在java测添加的一个锁机制实现的。

* FtpToHive

1. 该传输任务有点复杂。

首先根据配置，获取ftp上的文件列表，并根据文件名称的匹配列出ftp上所有需要下载的文件。

ftp支持配置多个路径，ftp的路径和文件名支持使用时间进行动态替换。

列出所有的文件后，判断当前任务的全局变量fileNameList是否为空，为空则将列出来的文件保存在该list中，根据配置项“datatransfer.download.maximum.number.files”配置的数量，一次下载指定数量的文件，下载完成后解压并上传到hdfs。

之后重复上述操作，直到文件全部下载并上传到hdfs。

该步骤中存在以下问题：每次重复都会重复列出需要下载的文件，但是只有第一次列出的的会保存在全局变量fileNameList中，并且下载完成的从中移除，后续重新列出文件的时候不会再保存到fileNameList中，也就是说，如果ftp上的文件超过一次最大下载数，每次下载都会存在一个列出所有需要下载的文件的无效操作(不知道该操作的目的，如果后续存在优化需求，建议进行优化，之前有询问过该处问题，据说之前这样处理是为了支持一边上传文件一边处理，事实上通过代码走读，没有这种功能)。

文件保存在本地的临时目录为：${%DTM\_HOME%}/ftp\_temp\_path/${传输任务task id}, 每次完成一批量的数据下载上传后会删除临时目录。

文件解压失败后备份的路径由系统配置项ftptohive.errorfile进行配置，路径为

${“ftptohive.errorfile”}/${传输任务job id}

默认值：${%DTM\_HOME%}/workdata/errorfile

文件备份完成后会删除原文件（注意，如果备份后原文件删除失败，可能会导致文件上传hdfs失败，代码中匹配文件并上传的时候如果有压缩文件没有删除会存在数组越界异常）



文件上传到hdfs的目录为：/transferInfo/ftptohive/input/${传输任务job id}

文件支持一下类型：txt/csv/tsv/gz/tr.gz 其中后两种文件需要解压缩，默认的文件编码格式为

1. 通过MapReduce任务，将文件数据通过指定函数转换后保存到hdfs上的输出文件夹中。该步骤主要讲输入字段按照配置的函数进行转换并且按照hive表的顺序进行排序输出到指定文件夹中，默认文件名称part-r-00000。

Job 的名称为ftptohive\_${传输任务job id}

执行任务时配置文件（用于配置输入输出字段对应关系）存放路径：/transferInfo/${transfertype}/config/${传输任务job id}/file\_conf.txt

输出文件夹路径：/transferInfo/${transfertype}/output/${传输任务job id}

执行job用到的MR jar包：datatransfer.jar

1. 根据所入的hive表是否为分区表决定不同的后续操作，不是分区表直接将输出目录中的文件保存到hive表中

Load文件进入hive表的语句示例：

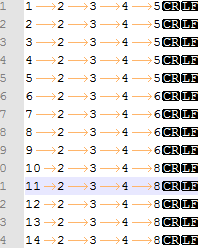
LOAD DATA INPATH '/transferInfo/${transfertype}/output/100180205/part-r-00000' INTO TABLE fff

将指定文件part-r-00000导入对应表中。

1. 如果hive 表为分区表，则需要做如下操作：

执行MapReduce任务，根据之前的输出文件，和hive表的分区字段，生成分区字段文件，这里不太好描述，举个例子

输出文件里面有5个字段对应要保存到hive表中，其中第4,5两个字段为分区字段，这时候要将4,5两个字段所有的排列取值列出来



如果输入文件如上图所示（注意，上述箭头为tab，是目前hive表的默认分隔符），输出文件为



也就是说分区字段上存以上三种排列组合，这个工作是为了后续能将文件按照上述分区字段分解成三个文件。

生成的文件保存在如下路径中：

/transferInfo/${transfertype}/parartion/${传输任务job id}

1. 根据上述的文件，为hive表创建分区目录

该步骤操作为直连hive数据库，执行hsql创建分区

示例语句：

alert table ${表名} add if not exists partition (${列名1}= ‘${列1的取值}’,${列名2}= ‘${列2的取值}’)

注意：

1．目前传输任务hive表只支持两层分区。

2．向分区表插入数据时，分区字段的排列组合不能过多。因为该表和oracle表类似，都是需要创建对应的分区再储存数据。

1. 通过Mapreduce，根据分区字段组合将之前生成的文件分割成若干小文件：

任务名称：

hivetoftp\_split\_data\_by\_partition\_column\_${传输任务job id}

文件的输出目录为：

/transferInfo/${transfertype}/output/parartion\_temp/${传输任务job id}

执行任务依赖的jar包：

SplitDirctoryFile.jar

根据MR任务原理，如果有多个输出文件需要一开始就确定输出文件个数，然后才能制定对应的数据输出，所以之前的确定分区排列组合的任务是必须的。

1. 最后将上述的输出文件load进入hive表对应的表空间中，和非分区表有略微的不同，这里将数据保存到hive表中需要指定对应的分区。

Hsql语句样例：

1. tips：hivetoftp传输任务应该属于dtm最开始做的一批传输任务，该流程奠定了整个dtm模块的传输任务如何将数据保存到hive表中的基础。其中涉及到三个MapReduce流程。

* OracleToHive

1. 根据oracle表数据的导出条件，从oracle表中查询总共存在多少条数据，如果数据条数为0，则直接返回告警状态（一般指数据为空的状态）
2. 根据配置项dtm.csv.linenum确定每次从hive表中查询出来的数据条数，分批次从oracle表中查询出数据，并生产csv文件，实际的执行效果是如果配置项的值为10，则每次从oracle表中查询出来10条数据，然后打开一个文件流，创建一个csv文件，文件为：${%DTM\_HOME%}/ftp\_temp\_path/${传输任务job id}/${oracleTableName}\_${i}.csv

（这里存在效率问题，查询出来的数据，和生成一个csv文件里面的数据使用的是用一个配置项控制，如果该配置项太小，我们会有太多次数据库交互，和多个csv文件。

正确的方式应该是分别控制，一次从数据库查询出来的数据条数在200-500条之间。查询出来的文件总条数在5000-10000条时，生成一个文件。用两个配置项，两个循环来控制文件的查询输出，比较合理）

1. 之后流程见ftptohive流程的b)🡪h)。

* LocalToHive

1. 首先是获取上传好的文件，从本地上传文件时，在创建任务的过程中就会对文件做一次初步的处理，实际处理的文件是通过初步解析以后重新生成的一个文件。
2. 之后流程见ftptohive流程的b)🡪h)。

* EmailToOracle

1. 从邮箱中查询符合规则的文件，下载到本地，下载完成后需要对文件预处理，如果文件编码格式为GBK，则在java测重新读写，将文件编码格式转换成UTF-8（这里之前做的时候存在问题，当时在hive上跑的MapReduce流程并不支持gbk格式的编码，因此是在java测做的转换，优化的话可以按照ftptohive的中gbk文件的处理方式进行修改）
2. 将预处理过的文件上传到hdfs指定目录

目录为hdfs上的input目录：

/transferInfo/${transfertype}/input/${传输任务job id}

1. 执行MapReduce任务，处理input目录的文件，将处理后的文件输出到hdfs上的如下目录

/transferInfo/${transfertype}/output/${传输任务job id}

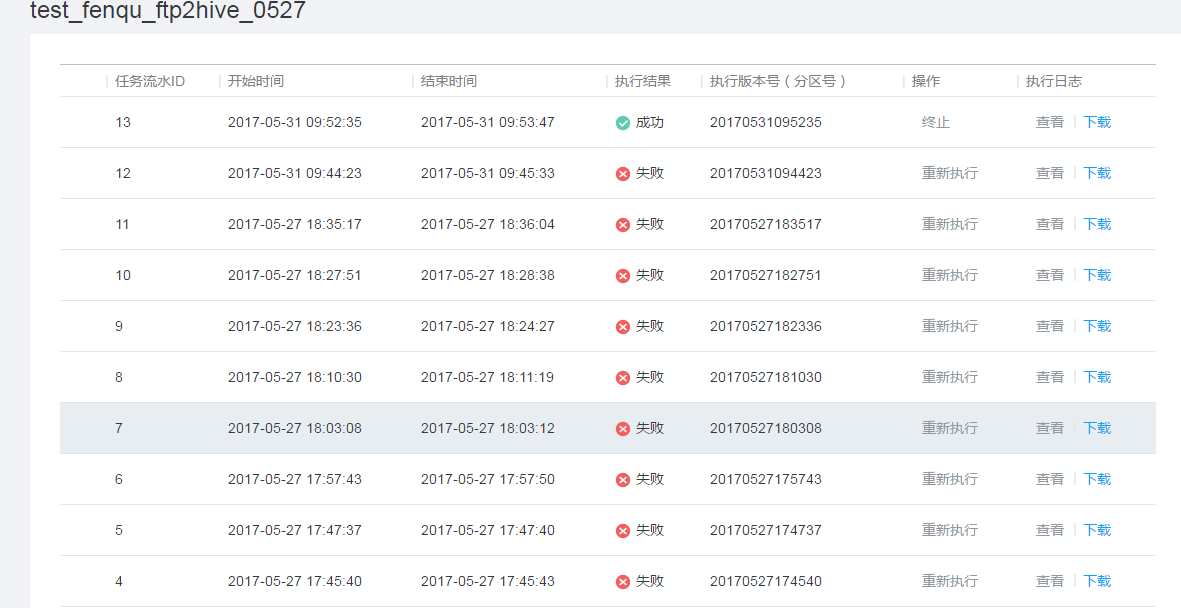
1. 读取输出结果到本地，链接指定oracle数据库，将数据数据结果保存到oracle数据库中，数据存入oracle数据库之前需要进行对每个字段依据oracle表的字段进行校验，校验不通过的数据舍弃，使用的是batchupdate，不支持事务操作。
2. 注意：当一个传输任务实例使用了一个oracle链接以后，其他的传输任务是不能使用该oracle表的，这里是在java测添加的一个锁机制实现的。

二．传输任务的其余相关信息

1. 数据传输任务的日志查看以及历史任务信息相关



1. 点开传输任务历史信息可以查看已经生成的传输任务的实例



1. 关于任务的执行结果

有成功/失败/执行中/待执行/告警五中，

其中告警结果对应的传输任务的源数据为空，比如ftptohive，从ftp没有获取到指定文件。

待执行是由于传输任务创建完成以后加载到java虚拟机的定时器中，定期创建任务实例，但是威力避免任务相互影响，同一时间只允许一个任务实例执行，如果一个任务配置的执行间隔为1分钟，而一分钟之内，第一个任务实例还在执行状态，则第二个任务不会执行，处于待执行状态，直到第一个任务执行完成。

1. 关于宕机重启

宕机重启后，宕机前执行中的传输任务状态会改为执行失败，宕机前待执行的传输任务实例会加载到内存中等待继续执行。

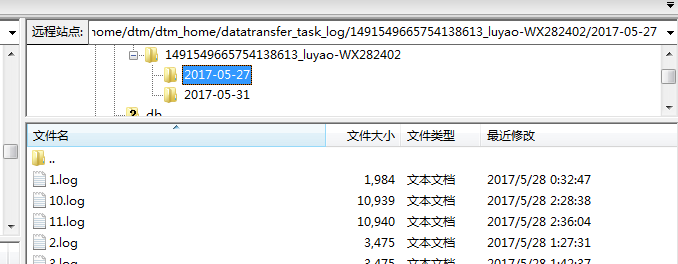
1. 关于传输任务执行日志

执行中的传输任务实例，可以通过执行日志栏的查看进行日志查看操作，由前台定时器定时（10s）向后台发送请求达到查看实时日志的目的，日志最多显示最新的300行，执行完成之后可以过下载按钮下载查看日志文件。

日志中主要打印了传输任务的链接信息，主流程的日志信息，以及主要的异常信息，可以模糊定位出传输任务执行到哪一步骤中发生问题。

传输任务日志存放路径：

${DTM\_HOME}/datatransfer\_task\_log/${userid}/${日期}/${传输任务job id}.log

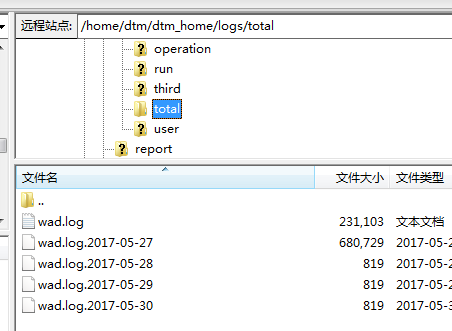


1. 深层次日志

如果传输任务日志不够全面定位不清楚，可以查看后台日志：后台日志存放路径：

${DTM\_HOME}/logs/total/wad.log

早期日志定位请找到对应日期的日志搜索传输任务job id进行查看



1. Hadoop yarn日志

如果是传输任务执行时候MRjob出现异常，则需要登陆hadoop yarn查看日志

例如，当前c60环境的hadoop yarn地址：[https://100.107.214.39:28443/web/index.html](https://100.107.214.39:28443/web/index.html?)

用户名/密码:admin/Wad\_1234

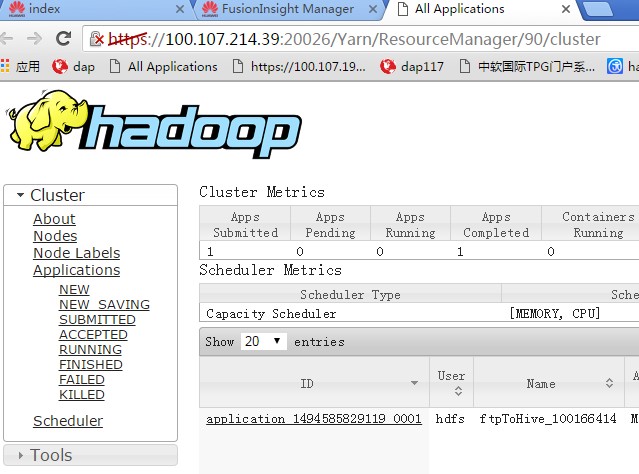
进入后从左边的软件栏中找到yarn点击进入，然以后点击下图框住部分，进入hadoop yarn。



进入hadoop yarn 如下图，一般你的传输任务在accepted/running/finished中查找，MR任务异常主要有任务阻塞（任务太多等待执行），任务缺少jar包（主要是缺少bsh.jar，新搭环境问题），以及其他执行中可能产生的问题。

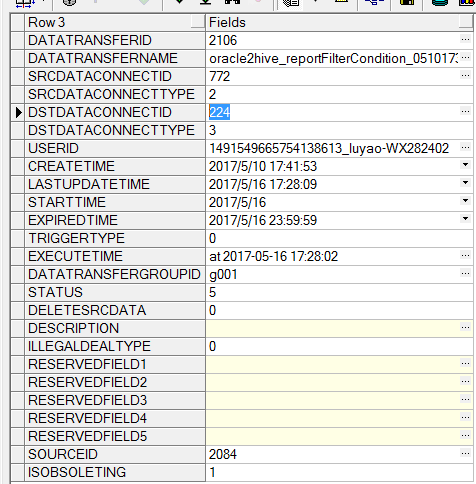
根据传输任务job id关键字，可以找到对应的MapReduce任务。

点击任务ID进入任务，找到里面的log，可以查看具体问题。



1. 传输任务的删除，正常传输任务删除需要删除数据库中各个表中对应该传输任务的信息，存在问题，修改后的传输任务的历史传输任务是没有正常删除的，这个问题是传输任务支持修改后，由于考虑不足而产生的，近期整理文档的时候才发现。

如下图所示的传输任务：修改最后一次的传输任务id是2106，传输任务依赖的基础是2084，应该先通过2106找到2084，然后在找到所有依赖与任务id2084的传输任务，逐一删除，实际上，删除的时候只删除了任务id2106的相关任务。



1. 关于dtm模块的java代码对应功能简介：

1.系统启动的时候初始化类SysInitServlet



这里的逻辑主要是系统配置项加载，包括数据库配置，xml文件配置，log4j文件配置，，定时任务的定时器加载。

2．数据传输任务定时器管理类DataTransferTimerManager



系统启动的时候加载定时器任务管理类，实现传输任务初始化

1. 将停机之前状态正在执行的传输任务实例置为失败状态。
2. 将停机之前状态为待执行状态的传输任务实例加载到缓存中立即执行
3. 获取未过期的定时任务信息，加载定时器，创建定时任务。

3．数据传输任务类DataTransferTask



该类的主要作用是定时生成定时任务实例，加载到定时任务控制器DataTransferContral中，控制器中通过一个线程安全的ConcurrentHashMap<String, LinkedBlockingQueue<String>>（该map中key保存定时任务id，value保存定时任务实例id）保存传输任务和传输任务实例，通过执行回调机制保证定时任务同一时间只有一个实例在执行。

4. 数据传输任务执行线程类DataTransferRunner。

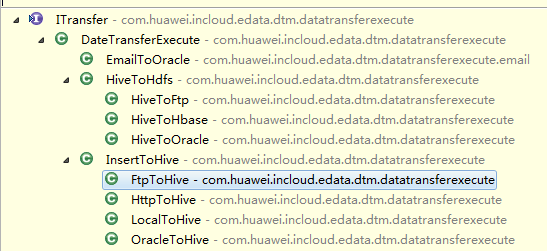
该类的主要作用是供定时器DataTransferContral调用，创建传输任务实例，并在传输任务实例执行完成之后回调定时器，触发下一个任务实例执行。

注意：这个线程类是直接调用的，没有线程数上限，可以考虑使用线程池调用，限制同时执行的定时任务上限。

5.传输任务实例创建工厂类DataTransferFactory，

6.用于创建传输任务类实例，并初始化。

传输任务执行类



ITransfer为传输任务类的接口

DataTransferExecute为传输任务实现类父类

所有的传输任务基础该父类

传输任务分为三类hivetoXXX/XXXtohive/EmailtoOracle

所有的hivetoXXX传输任务继承hivetohdfs操作

所有的XXXtohive传输任务继承InsertToHive操作

EmailtoOracle自己作为一类传输任务

为了减少代码的重复率，这里使用了继承的关系，把大部分的公共代码提取到父类中间，完成代码的优化操作。

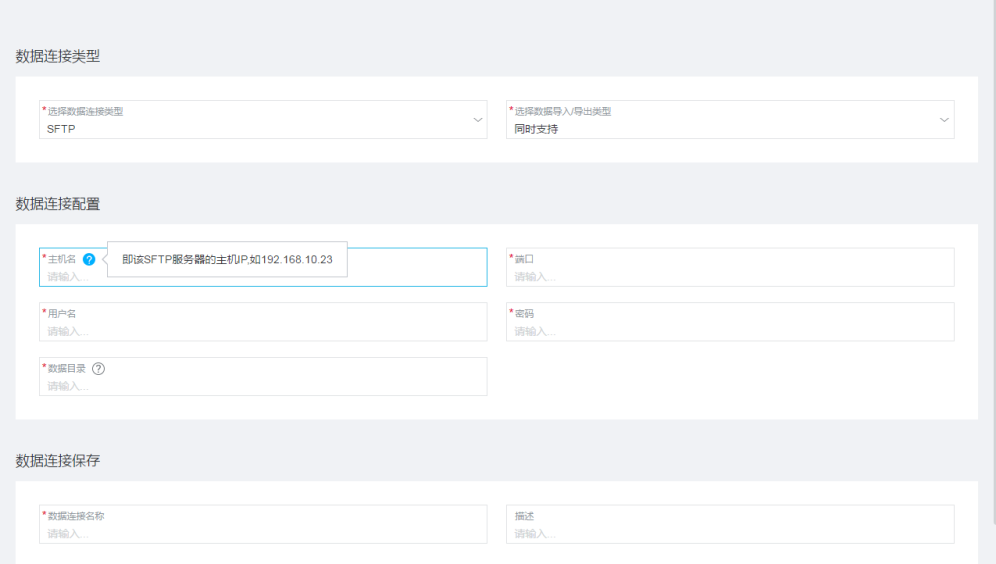
1. 各类型链接的创建以及相关配置信息。

关于数据链接的创建

数据链接支持如下各类型的数据链接创建：

ftp/hbase/hdfs/hive/http/oracle/sftp

各类型链接创建的时候页面给的提示很明确，可以直接按照提示创建

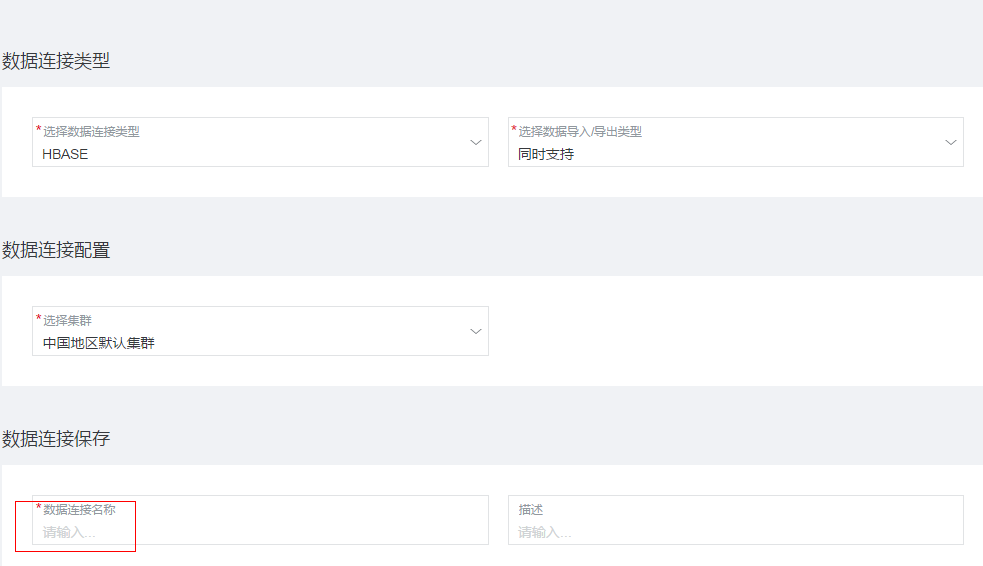


需要注意的是hbase链接。

Hbase/hive和链接只需要填写链接名称就可以了，实际上hbsae链接就是我们默认登陆hadoop的链接，创建一个实例即可使用。

链接一样如何区分不同用户创建的空间和表呢？

这里我传输任务中是根据用户名到后台跳转到dam模块去查询相关的数据库以及表信息的。



1. 关于dam模块hive和hbase的相关信息

Hive表和hbase表的创建需要在dap/dam模块进行。

1. 表的创建以及相关信息介绍。

表创建的页面在dap的数据集管理模块：支持关系型（指hive表）/nosql型（指hbase表）



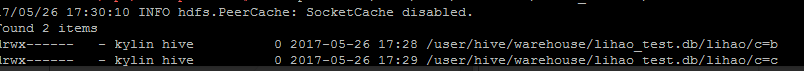
创建hive表的时候需要先创建数据库，再创建表，如下图所示：

创建数据库，再在数据库中创建表。





创建完以后hive表空间信息在hadoop中的文件路径如下图所示：

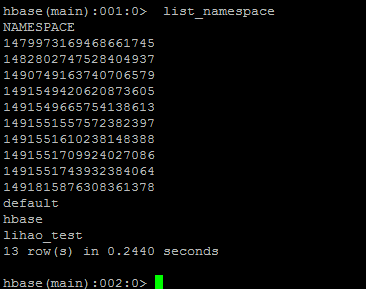


1. 关于hbase表的创建

对于hbase表来说，没有数据库这个概念，只有表空间，默认为default表空间，不支持新建。



1. 初始版本中所有的用户都在default表空间创建表，并且为用户创建表之后我们在dam 测的oracle数据库中记录了用户创建的hbase表信息，表名称即为用户创建的表名，实际上，在hbase表的default表空间中实际创建的表名称为${用户id}\_表名，比如用户为1479976440194646157\_luyao-WX282402，该用户创建的表名为aaa，那么在hbase中的实际表名称为WX282402\_aaa,下图为hbase中的表空间信息。之前创建的表名称为WX282402\_aaa的表就保存在下图中default表空间中。



1. 作为hbase，默认只有default和hbase表空间，为什么上述有那么多表空间，是应为之后版本有个需求，需要所有用户都拥有一个独立的表空间，表空间以用户唯一id命名，之后我用户1479976440194646157\_luyao-WX282402创建一个表bbb，实际表在1479976440194646157表空间中，表名称为bbb，以上的改变仅针对用户创建的表在hbase中的信息，对于前台的展示也就不做变化，这也是为了兼容之前版本做的一些考虑。

