尚硅谷大数据技术之 DataX

(作者：尚硅谷大数据研发部)

版本：V1.0

第 1 章 DataX 简介

1.1 DataX 概述

DataX 是阿里巴巴开源的一个异构数据源离线同步工具，致力于实现包括关系型数据库(MySQL、Oracle 等)、HDFS、Hive、ODPS、HBase、FTP 等各种异构数据源之间稳定高效的数据同步功能。

源码地址：https://github.com/alibaba/DataX

1.2 DataX 支持的数据源

DataX 目前已经有了比较全面的插件体系，主流的 RDBMS 数据库、NOSQL、大数据计算系统都已经接入，目前支持数据如下图。

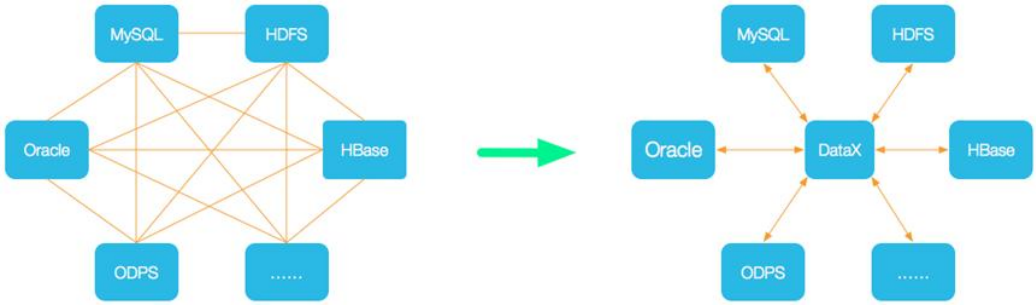
类型	数据源	Reader(读)	Writer(写)
RDBMS 关系型数据库	MySQL	√	√
	Oracle	√	√
	OceanBase	√	√
	SQLServer	√	√
	PostgreSQL	√	√
	DRDS	√	√
	通用 RDBMS	√	√
阿里云数仓数据存储	ODPS	√	√
	ADS		√
	OSS	√	√
	OCS	√	√
NoSQL 数据存储	OTS	√	√
	Hbase0.94	√	√
	Hbase1.1	√	√
	Phoenix4.x	√	√
	Phoenix5.x	√	√
	MongoDB	√	√
	Hive	√	√
	Cassandra	√	√
无结构化数据存储	TxtFile	√	√
	FTP	√	√

	HDFS	√	√
	Elasticsearch		√
时间序列数据库	OpenTSDB	√	
	TSDB	√	√

第 2 章 DataX 架构原理

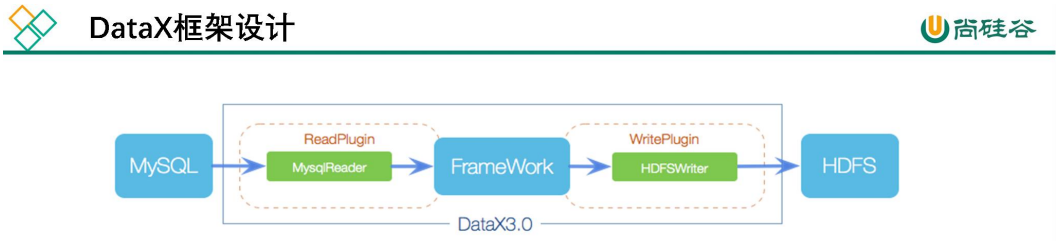
2.1 DataX 设计理念

为了解决异构数据源同步问题，DataX 将复杂的网状的同步链路变成了星型数据链路，DataX 作为中间传输载体负责连接各种数据源。当需要接入一个新的数据源的时候，只需要将此数据源对接到 DataX，便能跟已有的数据源做到无缝数据同步。



2.2 DataX 框架设计

DataX 本身作为离线数据同步框架，采用 Framework + plugin 架构构建。将数据源读取和写入抽象成为 Reader/Writer 插件，纳入到整个同步框架中。



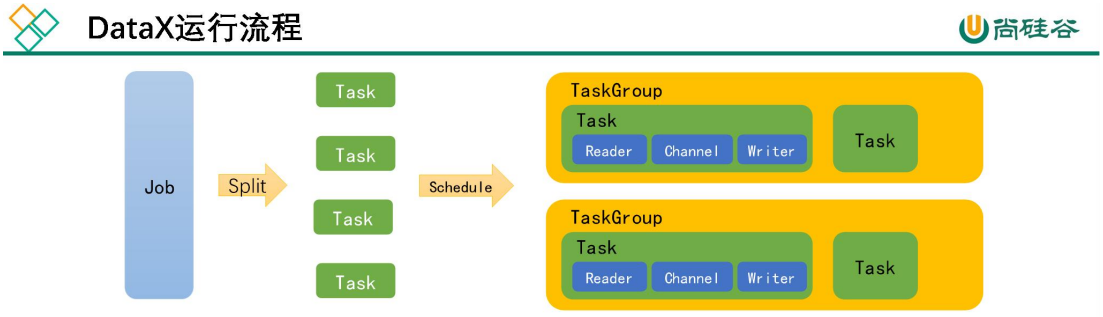
Reader：数据采集模块，负责采集数据源的数据，将数据发送给Framework。

Writer：数据写入模块，负责不断向Framework取数据，并将数据写入到目的端。

Framework：用于连接reader和writer，作为两者的数据传输通道，并处理缓冲，流控，并发，数据转换等核心技术问题。

2.3 DataX 运行流程

下面用一个 DataX 作业生命周期的时序图说明 DataX 的运行流程、核心概念以及每个概念之间的关系。



Job：单个数据同步的作业，称为一个Job，一个Job启动一个进程。

Task：根据不同数据源的切分策略，一个Job会切分为多个Task，Task是DataX作业的最小单元，每个Task负责一部分数据的同步工作。

TaskGroup：Scheduler调度模块会对Task进行分组，每个Task组称为一个Task Group。每个Task Group负责以一定的并发度运行其所分得的Task，单个Task Group的并发度为5。

Reader→Channel→Writer：每个Task启动后，都会固定启动Reader→Channel→Writer的线程来完成同步工作。

2.4 DataX 调度决策思路

举例来说，用户提交了一个 DataX 作业，并且配置了总的并发度为 20，目的是对一个有 100 张分表的 mysql 数据源进行同步。DataX 的调度决策思路是：

1）DataX Job 根据分库分表切分策略，将同步工作分成 100 个 Task。

2）根据配置的总的并发度 20，以及每个 Task Group 的并发度 5，DataX 计算共需要分配 4 个 TaskGroup。

3）4 个 TaskGroup 平分 100 个 Task，每一个 TaskGroup 负责运行 25 个 Task。

2.5 DataX 与 Sqoop 对比

功能	DataX	Sqoop
运行模式	单进程多线程	MR
分布式	不支持，可以通过调度系统规避	支持
流控	有流控功能	需要定制
统计信息	已有一些统计，上报需定制	没有，分布式的数据收集不方便
数据校验	在 core 部分有校验功能	没有，分布式的数据收集不方便
监控	需要定制	需要定制

第 3 章 DataX 部署

1）下载 DataX 安装包并上传到 hadoop102 的/opt/software

下载地址：http://datax-opensource.oss-cn-hangzhou.aliyuncs.com/datax.tar.gz

2）解压 datax.tar.gz 到/opt/module

```
[atguigu@hadoop102 software]$ tar -zxvf datax.tar.gz -C /opt/module/
```

3）自检，执行如下命令

```
[atguigu@hadoop102 ~]$ python /opt/module/datax/bin/datax.py /opt/module/datax/job/job.json
```

出现如下内容，则表明安装成功

```
……
2021-10-12 21:51:12.335 [job-0] INFO  JobContainer -
任务启动时刻                    : 2021-10-12 21:51:02
任务结束时刻                    : 2021-10-12 21:51:12
任务总计耗时                    :                 10s
任务平均流量                    :          253.91KB/s
记录写入速度                    :          10000rec/s
读出记录总数                    :              100000
读写失败总数                    :                   0
```

第 4 章 DataX 使用

4.1 DataX 使用概述

4.1.1 DataX 任务提交命令

DataX 的使用十分简单，用户只需根据自己同步数据的数据源和目的地选择相应的 Reader 和 Writer，并将 Reader 和 Writer 的信息配置在一个 json 文件中，然后执行如下命令提交数据同步任务即可。

```
[atguigu@hadoop102 datax]$ python bin/datax.py path/to/your/job.json
```

4.2.2 DataX 配置文件格式

可以使用如下命名查看 DataX 配置文件模板。

```
[atguigu@hadoop102 datax]$ python bin/datax.py -r mysqlreader -w hdfswriter
```

配置文件模板如下，json 最外层是一个 job，job 包含 setting 和 content 两部分，其中 setting 用于对整个 job 进行配置，content 用户配置数据源和目的地。

Reader 和 Writer 的具体参数可参考官方文档，地址如下：

https://github.com/alibaba/DataX/blob/master/README.md

https://gitee.com/mirrors/DataX/blob/master/README.md

类型	数据源	Reader(读)	Writer(写)	文档
RDBMS 关系型数据库	MySQL	√	√	读、写
	Oracle	√	√	读、写
	OceanBase	√	√	读、写
	SQLServer	√	√	读、写
	PostgreSQL	√	√	读、写
	DRDS	√	√	读、写
	通用RDBMS(支持所有关系型数据库)	√	√	读、写
阿里云数仓数据存储	ODPS	√	√	读、写
	ADS		√	写
	OSS	√	√	读、写
	OCS	√	√	读、写
NoSQL数据存储	OTS	√	√	读、写
	Hbase0.94	√	√	读、写
	Hbase1.1	√	√	读、写
	Phoenix4.x	√	√	读、写

4.2 同步 MySQL 数据到 HDFS 案例

案例要求：同步 gmall 数据库中 base_province 表数据到 HDFS 的/base_province 目录

需求分析：要实现该功能，需选用 MySQLReader 和 HDFSWriter，MySQLReader 具有两种模式分别是 TableMode 和 QuerySQLMode，前者使用 table，column，where 等属性声明需要同步的数据；后者使用一条 SQL 查询语句声明需要同步的数据。

下面分别使用两种模式进行演示。

4.2.1 MySQLReader 之 TableMode

1）编写配置文件

（1）创建配置文件 base_province.json

```
[atguigu@hadoop102 ~]$ vim /opt/module/datax/job/base_province.json
```

（2）配置文件内容如下

```
{
    "job": {
        "content": [
            {
                "reader": {
                    "name": "mysqlreader",
                    "parameter": {
                        "column": [
                            "id",
                            "name",
                            "region_id",
                            "area_code",
                            "iso_code",
                            "iso_3166_2"
                        ],
                        "where": "id>=3",
                        "connection": [
                            {
                                "jdbcUrl": [
                                    "jdbc:mysql://hadoop102:3306/gmall"
                                ],
                                "table": [
                                    "base_province"
                                ]
                            }
                        ],
                        "password": "123456",
                        "splitPk": "",
                        "username": "root"
                    }
                },
```

```
                    "writer": {
                        "name": "hdfswriter",
                        "parameter": {
                            "column": [
                                {
                                    "name": "id",
                                    "type": "bigint"
                                },
                                {
                                    "name": "name",
                                    "type": "string"
                                },
                                {
                                    "name": "region_id",
                                    "type": "string"
                                },
                                {
                                    "name": "area_code",
                                    "type": "string"
                                },
                                {
                                    "name": "iso_code",
                                    "type": "string"
                                },
                                {
                                    "name": "iso_3166_2",
                                    "type": "string"
                                }
                            ],
                            "compress": "gzip",
                            "defaultFS": "hdfs://hadoop102:8020",
                            "fieldDelimiter": "\t",
                            "fileName": "base_province",
                            "fileType": "text",
                            "path": "/base_province",
                            "writeMode": "append"
                        }
                    }
                }
            ],
            "setting": {
                "speed": {
                    "channel": 1
                }
            }
        }
}
```

2）配置文件说明

（1）Reader 参数说明

MySQLReader-TableMode

```
{
    "name": "mysqlreader",  Reader名称，固定写法
    "parameter": {
        "username": "root",    数据库用户名
        "password": "123456",  数据库密码
        "connection": [
            {
                "jdbcUrl": ["jdbc:mysql://hadoop102:3306/gmall"],  数据库JDBC URL
                "table": ["base_province"]   需要同步的表名
            }
        ],
        "column": ["id", "name", "region_id", "area_code", "iso_code", "iso_3166_2"],  需要同步的列，["*"]代表所有列
        "where": "id>=3",  where过滤条件
        "splitPk": ""   分片字段，如果指定该字段，则DataX会启动多个Task同步数据；若未指定(不提供
                        splitPk或者splitPk值为空)，则只会有有单个Task。该参数只在TableMode下有效，
                        意味着在QuerySQLMode下，只会有单个Task。Mysql一般切为5倍的并发数。
    }
}
```


（2）Writer 参数说明

HFDSWriter

```
    {
        "name": "hdfswriter",  Writer名称，固定写法
        "parameter": {
            "column": [          列信息，包括列名和类型。类型为Hive表字段类型，目前不支持decimal、binary、
                {                arrays、maps、structs等类型。若MySQL数据源中包含decimal类型字段，此处可
                    "name": "id",  将该字段类型设置为string，hive表中仍设置为decimal类型
                    "type": "bigint"
                }, {
                    "name": "name",
                    "type": "string"
                },
                ......
            ],
            "defaultFS": "hdfs://hadoop102:8020",  HDFS文件系统namenode节点地址
            "path": "/base_province",    HDFS文件系统目标路径
            "fileName": "base_province",  HDFS文件名前缀
            "fileType": "text",   HDFS文件类型，目前支持"text"或"orc"
            "compress": "gzip",  HDFS压缩类型，text文件支持gzip、bzip2；orc文件支持有NONE、SNAPPY
            "fieldDelimiter": "\t",  HDFS文件字段分隔符
            "writeMode": "append"  数据写入模式，append：追加；nonConflict：若写入目录
        }                          有同名(前缀相同)文件，报错
    }
```


注意事项：

HFDS Writer 并未提供 nullFormat 参数：也就是用户并不能自定义 null 值写到 HFDS 文件中的存储格式。默认情况下，HFDS Writer 会将 null 值存储为空字符串（''），而 Hive 默认的 null 值存储格式为\N。所以后期将 DataX 同步的文件导入 Hive 表就会出现问题。

解决该问题的方案有两个：

一是修改 DataX HDFS Writer 的源码，增加自定义 null 值存储格式的逻辑，可参考 https://blog.csdn.net/u010834071/article/details/105506580。

二是在 Hive 中建表时指定 null 值存储格式为空字符串（''），例如：

```
DROP TABLE IF EXISTS base_province;
CREATE EXTERNAL TABLE base_province
(
    `id`            STRING COMMENT '编号',
    `name`          STRING COMMENT '省份名称',
    `region_id`     STRING COMMENT '地区 ID',
    `area_code`     STRING COMMENT '地区编码',
    `iso_code`      STRING COMMENT '旧版 ISO-3166-2 编码，供可视化使用',
    `iso_3166_2`    STRING COMMENT '新版 IOS-3166-2 编码，供可视化使用'
) COMMENT '省份表'
    ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
    NULL DEFINED AS ''
    LOCATION '/base_province/';
```

（3）Setting 参数说明

Setting

```
{
    "setting": {
        "speed": {  传输速度配置
            "channel": 1  并发数，最终的并发数并不一定是该值，后边章节会进行解释说明
        },
        "errorLimit": {  容错比例配置
            "record": 1,  错误条数上限，超出则任务失败
            "percentage": 0.02  错误比例上限，超出则任务失败
        }
    }
}
```


3）提交任务

（1）在 HDFS 创建/base_province 目录

使用 DataX 向 HDFS 同步数据时，需确保目标路径已存在

```
[atguigu@hadoop102 datax]$ hadoop fs -mkdir /base_province
```

（2）进入 DataX 根目录

```
[atguigu@hadoop102 datax]$ cd /opt/module/datax
```

（3）执行如下命令

```
[atguigu@hadoop102 datax]$ python bin/datax.py job/base_province.json
```

4）查看结果

（1）DataX 打印日志

```
2021-10-13 11:13:14.930 [job-0] INFO    JobContainer -
任务启动时刻                            : 2021-10-13 11:13:03
任务结束时刻                            : 2021-10-13 11:13:14
任务总计耗时                            :                 11s
任务平均流量                            :              66B/s
记录写入速度                            :              3rec/s
读出记录总数                            :                 32
读写失败总数                            :                  0
```

（2）查看 HDFS 文件

```
[atguigu@hadoop102 datax]$ hadoop fs -cat /base_province/* | zcat

3    山西    1    140000    CN-14    CN-SX
4    内蒙古    1    150000    CN-15    CN-NM
5    河北    1    130000    CN-13    CN-HE
6    上海    2    310000    CN-31    CN-SH
7    江苏    2    320000    CN-32    CN-JS
8    浙江    2    330000    CN-33    CN-ZJ
9    安徽    2    340000    CN-34    CN-AH
10    福建    2    350000    CN-35    CN-FJ
11    江西    2    360000    CN-36    CN-JX
12    山东    2    370000    CN-37    CN-SD
14    台湾    2    710000    CN-71    CN-TW
15    黑龙江    3    230000    CN-23    CN-HL
16    吉林    3    220000    CN-22    CN-JL
17    辽宁    3    210000    CN-21    CN-LN
18    陕西    7    610000    CN-61    CN-SN
19    甘肃    7    620000    CN-62    CN-GS
20    青海    7    630000    CN-63    CN-QH
21    宁夏    7    640000    CN-64    CN-NX
22    新疆    7    650000    CN-65    CN-XJ
23    河南    4    410000    CN-41    CN-HA
24    湖北    4    420000    CN-42    CN-HB
25    湖南    4    430000    CN-43    CN-HN
26    广东    5    440000    CN-44    CN-GD
27    广西    5    450000    CN-45    CN-GX
28    海南    5    460000    CN-46    CN-HI
29    香港    5    810000    CN-91    CN-HK
30    澳门    5    820000    CN-92    CN-MO
31    四川    6    510000    CN-51    CN-SC
32    贵州    6    520000    CN-52    CN-GZ
33    云南    6    530000    CN-53    CN-YN
13    重庆    6    500000    CN-50    CN-CQ
34    西藏    6    540000    CN-54    CN-XZ
```

4.2.2 MySQLReader 之 QuerySQLMode

1）编写配置文件

（1）修改配置文件 base_province.json

```
[atguigu@hadoop102 ~]$ vim /opt/module/datax/job/base_province.json
```

（2）配置文件内容如下

```
{
    "job": {
        "content": [
            {
                "reader": {
                    "name": "mysqlreader",
                    "parameter": {
                        "connection": [
                            {
                                "jdbcUrl": [
                                    "jdbc:mysql://hadoop102:3306/gmall"
                                ],
                                "querySql": [
                                    "select
id,name,region_id,area_code,iso_code,iso_3166_2 from base_province where id>=3"
                                ]
                            }
                        ],
                        "password": "123456",
                        "username": "root"
                    }
                },
                "writer": {
                    "name": "hdfswriter",
                    "parameter": {
                        "column": [
                            {
                                "name": "id",
                                "type": "bigint"
                            },
                            {
                                "name": "name",
                                "type": "string"
                            },
                            {
                                "name": "region_id",
                                "type": "string"
                            },
                            {
                                "name": "area_code",
                                "type": "string"
                            },
                            {
                                "name": "iso_code",
                                "type": "string"
```

```
                                },
                                {
                                    "name": "iso_3166_2",
                                    "type": "string"
                                }
                            ],
                            "compress": "gzip",
                            "defaultFS": "hdfs://hadoop102:8020",
                            "fieldDelimiter": "\t",
                            "fileName": "base_province",
                            "fileType": "text",
                            "path": "/base_province",
                            "writeMode": "append"
                        }
                    }
                }
            ],
            "setting": {
                "speed": {
                    "channel": 1
                }
            }
        }
}
```

2）配置文件说明

（1）Reader 参数说明

MySQLReader-QuerySQLMode 尚硅谷

```
{
  "name": "mysqlreader",  Reader名称，固定写法
  "parameter": {
    "username": "root",   数据库用户名
    "password": "123456",  数据库密码
    "connection": [
      {
        "jdbcUrl": [
          "jdbc:mysql://hadoop102:3306/gmall"   数据库JDBC URL
        ],
        "querySql": [
          "select id,name,region_id,area_code,iso_code,iso_3166_2   查询语句
           from base_province where id>=3"
        ]
      }
    ]
  }
}
```

让天下没有难学的技术

3）提交任务

（1）清空历史数据

```
[atguigu@hadoop102 datax]$ hadoop fs -rm -r -f /base_province/*
```

（2）进入 DataX 根目录

```
[atguigu@hadoop102 datax]$ cd /opt/module/datax
```

（3）执行如下命令

```
[atguigu@hadoop102 datax]$ python bin/datax.py job/base_province.json
```

4）查看结果

（1）DataX 打印日志

```
2021-10-13 11:13:14.930 [job-0] INFO    JobContainer -
任务启动时刻                            : 2021-10-13 11:13:03
任务结束时刻                            : 2021-10-13 11:13:14
任务总计耗时                            :                 11s
任务平均流量                            :              66B/s
记录写入速度                            :            3rec/s
读出记录总数                            :                 32
读写失败总数                            :                  0
```

（2）查看 HDFS 文件

```
[atguigu@hadoop102 datax]$ hadoop fs -cat /base_province/* | zcat

3    山西    1    140000    CN-14    CN-SX
4    内蒙古    1    150000    CN-15    CN-NM
5    河北    1    130000    CN-13    CN-HE
6    上海    2    310000    CN-31    CN-SH
7    江苏    2    320000    CN-32    CN-JS
8    浙江    2    330000    CN-33    CN-ZJ
9    安徽    2    340000    CN-34    CN-AH
10    福建    2    350000    CN-35    CN-FJ
11    江西    2    360000    CN-36    CN-JX
12    山东    2    370000    CN-37    CN-SD
14    台湾    2    710000    CN-71    CN-TW
15    黑龙江    3    230000    CN-23    CN-HL
16    吉林    3    220000    CN-22    CN-JL
17    辽宁    3    210000    CN-21    CN-LN
18    陕西    7    610000    CN-61    CN-SN
19    甘肃    7    620000    CN-62    CN-GS
20    青海    7    630000    CN-63    CN-QH
21    宁夏    7    640000    CN-64    CN-NX
22    新疆    7    650000    CN-65    CN-XJ
23    河南    4    410000    CN-41    CN-HA
24    湖北    4    420000    CN-42    CN-HB
25    湖南    4    430000    CN-43    CN-HN
26    广东    5    440000    CN-44    CN-GD
27    广西    5    450000    CN-45    CN-GX
28    海南    5    460000    CN-46    CN-HI
29    香港    5    810000    CN-91    CN-HK
30    澳门    5    820000    CN-92    CN-MO
31    四川    6    510000    CN-51    CN-SC
32    贵州    6    520000    CN-52    CN-GZ
```

```
33  云南    6   530000  CN-53   CN-YN
13  重庆    6   500000  CN-50   CN-CQ
34  西藏    6   540000  CN-54   CN-XZ
```

4.2.3 DataX 传参

通常情况下，离线数据同步任务需要每日定时重复执行，故 HDFS 上的目标路径通常会包含一层日期，以对每日同步的数据加以区分，也就是说每日同步数据的目标路径不是固定不变的，因此 DataX 配置文件中 HDFS Writer 的 path 参数的值应该是动态的。为实现这一效果，就需要使用 DataX 传参的功能。

DataX 传参的用法如下，在 JSON 配置文件中使用${param}引用参数，在提交任务时使用-p"-Dparam=value"传入参数值，具体示例如下。

1）编写配置文件

（1）修改配置文件 base_province.json

```
[atguigu@hadoop102 ~]$ vim /opt/module/datax/job/base_province.json
```

（2）配置文件内容如下

```
{
    "job": {
        "content": [
            {
                "reader": {
                    "name": "mysqlreader",
                    "parameter": {
                        "connection": [
                            {
                                "jdbcUrl": [
                                    "jdbc:mysql://hadoop102:3306/gmall"
                                ],
                                "querySql": [
                                    "select
id,name,region_id,area_code,iso_code,iso_3166_2 from base_province where id>=3"
                                ]
                            }
                        ],
                        "password": "123456",
                        "username": "root"
                    }
                },
                "writer": {
                    "name": "hdfswriter",
                    "parameter": {
                        "column": [
                            {
                                "name": "id",
```

```
                                "type": "bigint"
                            },
                            {
                                "name": "name",
                                "type": "string"
                            },
                            {
                                "name": "region_id",
                                "type": "string"
                            },
                            {
                                "name": "area_code",
                                "type": "string"
                            },
                            {
                                "name": "iso_code",
                                "type": "string"
                            },
                            {
                                "name": "iso_3166_2",
                                "type": "string"
                            }
                        ],
                        "compress": "gzip",
                        "defaultFS": "hdfs://hadoop102:8020",
                        "fieldDelimiter": "\t",
                        "fileName": "base_province",
                        "fileType": "text",
                        "path": "/base_province/${dt}",
                        "writeMode": "append"
                    }
                }
            }
        ],
        "setting": {
            "speed": {
                "channel": 1
            }
        }
    }
}
```

2）提交任务

（1）创建目标路径

```
[atguigu@hadoop102 datax]$ hadoop fs -mkdir /base_province/2020-06-14
```

（2）进入 DataX 根目录

```
[atguigu@hadoop102 datax]$ cd /opt/module/datax
```

（3）执行如下命令

```
[atguigu@hadoop102 datax]$ python bin/datax.py -p"-Ddt=2020-06-14" job/base_province.json
```

3）查看结果

```
[atguigu@hadoop102 datax]$ hadoop fs -ls /base_province
Found 2 items
drwxr-xr-x   - atguigu supergroup          0 2021-10-15 21:41 /base_province/2020-06-14
```

4.3 同步 HDFS 数据到 MySQL 案例

案例要求：同步 HDFS 上的/base_province 目录下的数据到 MySQL gmall 数据库下的 test_province 表。

需求分析：要实现该功能，需选用 HDFSReader 和 MySQLWriter。

1）编写配置文件

（1）创建配置文件 test_province.json

```
[atguigu@hadoop102 ~]$ vim /opt/module/datax/job/base_province.json
```

（2）配置文件内容如下

```
{
    "job": {
        "content": [
            {
                "reader": {
                    "name": "hdfsreader",
                    "parameter": {
                        "defaultFS": "hdfs://hadoop102:8020",
                        "path": "/base_province",
                        "column": [
                            "*"
                        ],
                        "fileType": "text",
                        "compress": "gzip",
                        "encoding": "UTF-8",
                        "nullFormat": "\\N",
                        "fieldDelimiter": "\t",
                    }
                },
                "writer": {
                    "name": "mysqlwriter",
                    "parameter": {
                        "username": "root",
                        "password": "123456",
                        "connection": [
                            {
                                "table": [
                                    "test_province"
                                ],
                                "jdbcUrl":
"jdbc:mysql://hadoop102:3306/gmall?useUnicode=true&characterEncoding=utf-8"
```

```
                                }
                            ],
                            "column": [
                                "id",
                                "name",
                                "region_id",
                                "area_code",
                                "iso_code",
                                "iso_3166_2"
                            ],
                            "writeMode": "replace"
                        }
                    }
                }
            ],
            "setting": {
                "speed": {
                    "channel": 1
                }
            }
        }
}
```

2）配置文件说明

（1）Reader 参数说明

HDFSReader 尚硅谷

```
{
    "name": "hdfsreader",    Reader名称，固定写法
    "parameter": {
        "defaultFS": "hdfs://hadoop102:8020",    HDFS文件系统namenode地址
        "path": "/base_province",    文件所在路径
        "column": [    需要同步的列，可使用索引选择所需列，例如[{ “index”: 0, “type”: “long” },
            "*"            { “index”: 1, “type”: “boolean” }]标识前两列，[“*”]标识所有列。
        ],
        "fileType": "text",    文件类型，目前支持textfile(text)、orcfile(orc)、rcfile(rc)、sequence file(seq)和csv文件(csv)
        "compress": "gzip",    压缩类型，目前支持gzip、bz2、zip、lzo、lzo_deflate、snappy等
        "encoding": "UTF-8",    文件编码
        "nullFormat": "\\N",    null值存储格式
        "fieldDelimiter": "\t"    字段分隔符
    }
}
```


（2）Writer 参数说明

```
{
    "name": "mysqlwriter",   Writer名称，固定写法
    "parameter": {
        "username": "root",    数据库用户名
        "password": "123456",  数据库密码
        "connection": [
            {
                "table": [
                    "test_province"    目标表
                ],
                "jdbcUrl": "jdbc:mysql://hadoop102:3306/gmall? JDBC URL
                  useUnicode=true&characterEncoding=utf-8"
            }
        ],
        "column": ["id","name", "region_id", "area_code", "iso_code", "iso_3166_2"],  目标列
        "writeMode": "replace"   写入方式：控制写入数据到目标表采用 insert into(insert) 或者 replace
                                 into(replace) 或者 ON DUPLICATE KEY UPDATE (update)语句
    }
}
```


3）提交任务

（1）在 MySQL 中创建 gmall.test_province 表

```
DROP TABLE IF EXISTS `test_province`;
CREATE TABLE `test_province`  (
  `id` bigint(20) NOT NULL,
  `name` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `region_id` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `area_code` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `iso_code` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `iso_3166_2` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci ROW_FORMAT = Dynamic;
```

（2）进入 DataX 根目录

```
[atguigu@hadoop102 datax]$ cd /opt/module/datax
```

（3）执行如下命令

```
[atguigu@hadoop102 datax]$ python bin/datax.py job/test_province.json
```

4）查看结果

（1）DataX 打印日志

```
2021-10-13 15:21:35.006 [job-0] INFO    JobContainer -
任务启动时刻                    : 2021-10-13 15:21:23
任务结束时刻                    : 2021-10-13 15:21:35
任务总计耗时                    :                 11s
```

```
任务平均流量                :          70B/s
记录写入速度                :          3rec/s
读出记录总数                :            34
读写失败总数                :             0
```

（2）查看 MySQL 目标表数据

id	name	region_id	area_code	iso_code	iso_3166_2
1	北京	1	110000	CN-11	CN-BJ
2	天津	1	120000	CN-12	CN-TJ
3	山西	1	140000	CN-14	CN-SX
4	内蒙古	1	150000	CN-15	CN-NM
5	河北	1	130000	CN-13	CN-HE
6	上海	2	310000	CN-31	CN-SH
7	江苏	2	320000	CN-32	CN-JS
8	浙江	2	330000	CN-33	CN-ZJ
9	安徽	2	340000	CN-34	CN-AH

第 5 章 DataX 优化

5.1 速度控制

DataX3.0 提供了包括通道(并发)、记录流、字节流三种流控模式，可以随意控制你的作业速度，让你的作业在数据库可以承受的范围内达到最佳的同步速度。

关键优化参数如下：

参数	说明
job.setting.speed.channel	总并发数
job.setting.speed.record	总 record 限速
job.setting.speed.byte	总 byte 限速
core.transport.channel.speed.record	单个 channel 的 record 限速，默认值为 10000（10000 条/s）
core.transport.channel.speed.byte	单个 channel 的 byte 限速，默认值 1024*1024（1M/s）

注意事项：

1.若配置了总 record 限速，则必须配置单个 channel 的 record 限速

2.若配置了总 byte 限速，则必须配置单个 channe 的 byte 限速

3.若配置了总 record 限速和总 byte 限速，channel 并发数参数就会失效。因为配置了总 record 限速和总 byte 限速之后，实际 channel 并发数是通过计算得到的：

计算公式为:

min(总 byte 限速/单个 channle 的 byte 限速，总 record 限速/单个 channel 的 record 限速)

配置示例：

```
{
```

```
    "core": {
        "transport": {
            "channel": {
                "speed": {
                    "byte": 1048576 //单个 channel byte 限速 1M/s
                }
            }
        }
    },
    "job": {
        "setting": {
            "speed": {
                "byte" : 5242880 //总 byte 限速 5M/s
            }
        },
        ...
    }
}
```

5.2 内存调整

当提升 DataX Job 内 Channel 并发数时，内存的占用会显著增加，因为 DataX 作为数据交换通道，在内存中会缓存较多的数据。例如 Channel 中会有一个 Buffer，作为临时的数据交换的缓冲区，而在部分 Reader 和 Writer 的中，也会存在一些 Buffer，为了防止 OOM 等错误，需调大 JVM 的堆内存。

建议将内存设置为 4G 或者 8G，这个也可以根据实际情况来调整。

调整 JVM xms xmx 参数的两种方式：一种是直接更改 datax.py 脚本；另一种是在启动的时候，加上对应的参数，如下：

```
python datax/bin/datax.py --jvm="-Xms8G -Xmx8G" /path/to/your/job.json
```