DataX

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

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

3）自检，执行如下命令

```
[iflytek@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

```
……
2024-7-12 21:51:12.335 [job-0] INFO  JobContainer -
任务启动时刻                    : 2024-7-12 21:51:02
任务结束时刻                    : 2024-7-12 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
```

DataX 使用案例（了解，不在项目中使用）

1 DataX 使用概述

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。

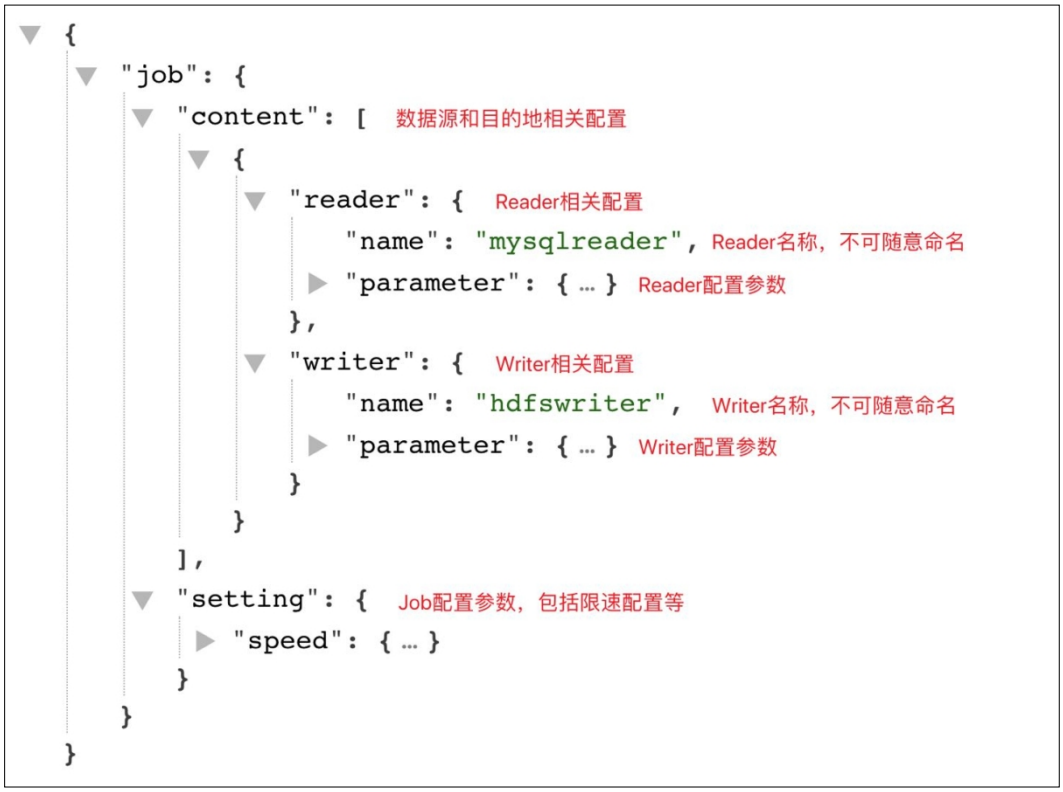
```
[iflytek@hadoop102 datax]$ python bin/datax.py path/to/your/job.json
```

1.2 DataX 配置文件格式

可以使用如下命名查看 DataX 配置文件模板。

```
[iflytek@hadoop102 software]$ cd /opt/module/datax/
[iflytek@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

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

2 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

```
[iflytek@hadoop102                    ~]$                    vim
/opt/module/datax/job/test_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": "000000",
                        "connection": [
                            {
                                "table": [
                                    "test_province"
                                ],
                                "jdbcUrl":
"jdbc:mysql://hadoop102:3306/gmall?useUnicode=true&allowPublicKey
Retrieval=true&characterEncoding=utf-8"
                            }
                        ],
                        "column": [
                            "id",
                            "name",
                            "region_id",
                            "area_code",
                            "iso_code",
                            "iso_3166_2",
                            "create_time",
```

```
                        "operate_time"
                    ],
                    "writeMode": "replace"
                }
            }
        }
    ],
    "setting": {
        "speed": {
            "channel": 1
        }
    }
  }
}
```

2）配置文件说明

（1）Reader 参数说明

```
{
  "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": "000000",   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,
  `create_time`    varchar(20)    CHARACTER    SET    utf8    COLLATE
utf8_general_ci NULL DEFAULT NULL,
  `operate_time`    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

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

（3）执行如下命令

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

4）查看结果

（1）DataX 打印日志

```
2024-7-13 15:21:35.006 [job-0] INFO  JobContainer -
任务启动时刻                    : 2024-7-13 15:21:23
任务结束时刻                    : 2024-7-13 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                    :                 32
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	create_time	operate_time
3	山西	1	140000	CN-14	CN-SX	2021-12-14 00:00:00	(Null)
4	内蒙古	1	150000	CN-15	CN-NM	2021-12-14 00:00:00	(Null)
5	河北	1	130000	CN-13	CN-HE	2021-12-14 00:00:00	(Null)
6	上海	2	310000	CN-31	CN-SH	2021-12-14 00:00:00	(Null)
7	江苏	2	320000	CN-32	CN-JS	2021-12-14 00:00:00	(Null)
8	浙江	2	330000	CN-33	CN-ZJ	2021-12-14 00:00:00	(Null)
9	安徽	2	340000	CN-34	CN-AH	2021-12-14 00:00:00	(Null)
10	福建	2	350000	CN-35	CN-FJ	2021-12-14 00:00:00	(Null)
11	江西	2	360000	CN-36	CN-JX	2021-12-14 00:00:00	(Null)
12	山东	2	370000	CN-37	CN-SD	2021-12-14 00:00:00	(Null)

Mysql 建表

在使用 DataX 导出 HDFS 数据之前，需要在 Mysql 建立对应的表，建表语句如下:

- **建立对应的库**

--建库

```
CREATE DATABASE IF NOT EXISTS medical_report DEFAULT CHARSET utf8
COLLATE utf8_general_ci;
```

- **创建省份表，并插入数据**

建表

```
DROP TABLE IF EXISTS `base_province`;
CREATE TABLE `base_province` (
  `id` bigint NOT NULL COMMENT 'id',
  `name`    varchar(20)    CHARACTER    SET    utf8mb3    COLLATE
utf8mb3_general_ci DEFAULT NULL COMMENT '省份名称',
  `region_id`    varchar(20)    CHARACTER    SET    utf8mb3    COLLATE
utf8mb3_general_ci DEFAULT NULL COMMENT '地区 id',
  `area_code`    varchar(20)    CHARACTER    SET    utf8mb3    COLLATE
utf8mb3_general_ci DEFAULT NULL COMMENT '地区编码',
  `iso_code`    varchar(20)    CHARACTER    SET    utf8mb3    COLLATE
utf8mb3_general_ci DEFAULT NULL COMMENT '旧版国际标准地区编码，供可视化
使用',
  `iso_3166_2`    varchar(20)    CHARACTER    SET    utf8mb3    COLLATE
utf8mb3_general_ci DEFAULT NULL COMMENT '新版国际标准地区编码，供可视化
使用',
  `create_time` datetime DEFAULT NULL COMMENT '创建时间',
  `operate_time` datetime DEFAULT NULL COMMENT '修改时间',
  PRIMARY KEY (`id`) USING BTREE
)   ENGINE=InnoDB   DEFAULT   CHARSET=utf8mb3   ROW_FORMAT=DYNAMIC
COMMENT='省份表';
```

数据插入

```
INSERT INTO `base_province` VALUES ('1', '北京市', '1', '110000',
'CN-11', 'CN-BJ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('2', '天津市', '1', '120000',
'CN-12', 'CN-TJ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('3', '山西省', '1', '140000',
'CN-14', 'CN-SX', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('4', '内蒙古', '1', '150000',
'CN-15', 'CN-NM', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('5', '河北省', '1', '130000',
'CN-13', 'CN-HE', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('6', '上海市', '2', '310000',
'CN-31', 'CN-SH', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('7', '江苏省', '2', '320000',
'CN-32', 'CN-JS', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('8', '浙江省', '2', '330000',
'CN-33', 'CN-ZJ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('9', '安徽省', '2', '340000',
'CN-34', 'CN-AH', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('10', '福建省', '2', '350000',
'CN-35', 'CN-FJ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('11', '江西省', '2', '360000',
'CN-36', 'CN-JX', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('12', '山东省', '2', '370000',
'CN-37', 'CN-SD', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('13', '重庆市', '6', '500000',
```

```
'CN-50', 'CN-CQ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('14', '台湾', '2', '710000',
'CN-71', 'CN-TW', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('15', '黑龙江省', '3', '230000',
'CN-23', 'CN-HL', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('16', '吉林省', '3', '220000',
'CN-22', 'CN-JL', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('17', '辽宁省', '3', '210000',
'CN-21', 'CN-LN', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('18', '陕西省', '7', '610000',
'CN-61', 'CN-SN', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('19', '甘肃省', '7', '620000',
'CN-62', 'CN-GS', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('20', '青海省', '7', '630000',
'CN-63', 'CN-QH', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('21', '宁夏', '7', '640000',
'CN-64', 'CN-NX', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('22', '新疆', '7', '650000',
'CN-65', 'CN-XJ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('23', '河南省', '4', '410000',
'CN-41', 'CN-HA', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('24', '湖北省', '4', '420000',
'CN-42', 'CN-HB', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('25', '湖南省', '4', '430000',
'CN-43', 'CN-HN', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('26', '广东省', '5', '440000',
'CN-44', 'CN-GD', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('27', '广西', '5', '450000',
'CN-45', 'CN-GX', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('28', '海南省', '5', '460000',
'CN-46', 'CN-HI', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('29', '香港', '5', '810000',
'CN-91', 'CN-HK', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('30', '澳门', '5', '820000',
'CN-92', 'CN-MO', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('31', '四川省', '6', '510000',
'CN-51', 'CN-SC', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('32', '贵州省', '6', '520000',
'CN-52', 'CN-GZ', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('33', '云南省', '6', '530000',
'CN-53', 'CN-YN', '2021-12-14 00:00:00', null);
INSERT INTO `base_province` VALUES ('34', '西藏', '6', '540000',
'CN-54', 'CN-XZ', '2021-12-14 00:00:00', null);
```

- **不同省份中医院数量建立 Mysql 表**

```
DROP TABLE IF EXISTS `province_medicine_hospital_num`;
CREATE TABLE `province_medicine_hospital_num` (
  `province` varchar(20) NOT NULL COMMENT '省份名称',
  `med_hos_num` bigint(20) NOT NULL COMMENT '中医药数量',
  `iso_3166_2` varchar(20) DEFAULT NULL COMMENT '地标',
  PRIMARY KEY (`province`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '不同省份中医院数量' ROW_FORMAT = DYNAMIC;
```

- **所有中医院不同职称医生的数量建立 Mysql 表**

```
DROP TABLE IF EXISTS `all_hospital_doctor_titile_num`;
CREATE TABLE `all_hospital_doctor_titile_num`  (
  `title` varchar(20) NOT NULL COMMENT '职称',
  `doctor_num` bigint(20) NOT NULL COMMENT '医生数量',
  PRIMARY KEY (`title`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '所有中医院不同职称医生数量' ROW_FORMAT = DYNAMIC;
```

- **每个类型的药品数量建立 Mysql 表**

```
DROP TABLE IF EXISTS `type_medicine_num`;
CREATE TABLE `type_medicine_num`  (
  `dose_type` varchar(20) NOT NULL COMMENT '药物分类',
  `medicine_num` bigint(20) NOT NULL COMMENT '药品种类数',
  PRIMARY KEY (`dose_type`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '每个类型的药品数量' ROW_FORMAT = DYNAMIC;
```

- **历史以来不同中医院的病人数建立 Mysql 表**

```
DROP TABLE IF EXISTS `history_medical_hospital_patient_num`;
CREATE TABLE `history_medical_hospital_patient_num`  (
  `name` varchar(20) NOT NULL COMMENT '中医院名称',
  `patient_num` bigint(20) NOT NULL COMMENT '病人数',
  PRIMARY KEY (`name`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '历史以来不同中医院的病人数' ROW_FORMAT = DYNAMIC;
```

- **历史以来不同中医院就诊次数建立 Mysql 表**

```
DROP TABLE IF EXISTS `history_medical_hospital_consultation_num`;
CREATE TABLE `history_medical_hospital_consultation_num`  (
  `name` varchar(20) NOT NULL COMMENT '中医院名称',
  `consultation_num` bigint(20) NOT NULL COMMENT '病人数',
  PRIMARY KEY (`name`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '历史以来不同中医院就诊次数' ROW_FORMAT = DYNAMIC;
```

- **需求 6 统计历史以来中医院收入 TOP 5**

```
DROP TABLE IF EXISTS `history_medical_hospital_amount_top5`;
CREATE TABLE `history_medical_hospital_amount_top5`  (
  `name` varchar(20) NOT NULL COMMENT '中医院名称',
  `total_amount` bigint(20) NOT NULL COMMENT '总金额',
  PRIMARY KEY (`name`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci
COMMENT = '历史以来中医院收入 TOP 5' ROW_FORMAT = DYNAMIC;
```

DataX 导出脚本使用

使用脚本导出 Hive 分析数据到 Mysql

1. 修改/opt/module/gen_datax_config/configuration.properties 文件

```
[iflytek@hadoop102 gen_datax_config]$ vim configuration.properties
```

文件内容修改如下。

```
mysql.username=root
mysql.password=000000
mysql.host=hadoop102
mysql.port=3306
```

```
mysql.database.import=medical
# 从 HDFS 导出进入的 MySQL 数据库名称
mysql.database.export=medical_report
#mysql.tables.import=
# MySQL 库中需要导出的表，空串表示导出库的所有表
mysql.tables.export=
is.seperated.tables=0
hdfs.uri=hdfs://hadoop102:8020
#import_out_dir=/opt/module/datax/job/import
# DataX 导出配置文件存放路径
export_out_dir=/opt/module/datax/job/export
```

2. 执行配置文件生成器

```
[iflytek@hadoop102 gen_datax_config]$ java -jar datax-config-generator-1.0-SNAPSHOT-jar-with-dependencies.jar
```

3. 进入目录查看脚本

```
[iflytek@hadoop102 gen_datax_config]$ cd /opt/module/datax/job/export
```

4. 查看生成 datax 脚本

```
medical_report.all_hospital_doctor_titile_num.json
medical_report.history_medical_hospital_amount_top5.json
medical_report.history_medical_hospital_consultation_num.json
medical_report.history_medical_hospital_patient_num.json
medical_report.province_medicine_hospital_num.json
medical_report.type_medicine_num.json
```

5. DataX 执行导出命令

不同省份中医院数量导出命令

执行命令之前修改改导出脚本

```
[iflytek@hadoop102 export]$ vim /opt/module/datax/job/export/medical_report.province_medicine_hospital_num.json
```

进入之后修改如下内容（删除 iso_3166_2 字段）

```
{"job":{"setting":{"speed":{"channel":1}},"content":[{"reader":{"name":"hdfsreader","parameter":{"path":"${exportdir}"
,"defaultFS":"hdfs://hadoop102:8020","column":["*"],"fileType":"text","encoding":"UTF-8","fieldDelimiter":"\t","nullFo
rmat":"\\N"}},"writer":{"name":"mysqlwriter","parameter":{"writeMode":"replace","username":"root","password":"000000",
"column":["province","med_hos_num","iso_3166_2"],"connection":[{"jdbcUrl":"jdbc:mysql://hadoop102:3306/medical_report?
useSSL=false&allowPublicKeyRetrieval=true&useUnicode=true&characterEncoding=utf-8","table":["province_medicine_hospita
l_num"]}]}}}]}
~
```

保存后执行导出命令

```
Python /opt/module/datax/bin/datax.py -p"-Dexportdir=/medical/province_medicine_hospital_num" /opt/module/datax/job/export/medical_report.province_medicine_hospital_num.json
```

所有中医院不同职称医生数量导出命令

```
python /opt/module/datax/bin/datax.py -p"-Dexportdir=/medical/all_hospital_doctor_titile_num" /opt/module/datax/job/export/medical_report.all_hospital_doctor_titile_num.json
```

每个类型的药品数量导出命令

```
python /opt/module/datax/bin/datax.py -p"-Dexportdir=/medical/type_medicine_num" /opt/module/datax/job/export/medical_report.type_medicine_num.json
```

历史以来不同中医院的病人数导出命令

```
python /opt/module/datax/bin/datax.py -p"-Dexportdir=/medical/history_medical_hospital_patient_num" /opt/module/datax/job/export/medical_report.history_medical_hospital_patien
```

```
t_num.json
```

历史以来不同中医院就诊次数导出命令

```
python                    /opt/module/datax/bin/datax.py                    -p"-
Dexportdir=/medical/history_medical_hospital_consultation_num"
/opt/module/datax/job/export/medical_report.history_medical_hospital_consul
tation_num.json
```

历史以来中医院收入 TOP 5 导出命令

```
python                    /opt/module/datax/bin/datax.py                    -p"-
Dexportdir=/medical/history_medical_hospital_amount_top5"
/opt/module/datax/job/export/medical_report.history_medical_hospital_amount
_top5.json
```

导出结果（以不同省份中医院数量为例）

province	med_hos_num
上海市	14
云南省	15
内蒙古	7
北京市	34
吉林省	26
四川省	32
天津市	11
宁夏	2
安徽省	29
山东省	70
山西省	15
广东省	71
广西	21

注意：在完成导出后，在 Mysql 语句中需要关联省份信息，更新 iso_3166_2

```
UPDATE province_medicine_hospital_num t1
LEFT JOIN base_province t2
ON t1.province = t2.`name`
SET t1.iso_3166_2 = t2.iso_3166_2;
```