尚硅谷电商用户画像四

版本：V 1.0

张晨

# 第七章 开发任务三：

# 数据迁移至Clickhouse

## 7.1 为何要迁移

标签计算完成后保存在hive虽然可以查询但是性能非常糟糕。而标签的使用往往是即时的。最常见的场景就是“用户分群”，也称“人群圈选”、“圈人”等等。

分群操作就是根据多个标签组合，产生一个用户集合，供营销、广告等部门使用。而这些操作计算量大，产生结果需要时效性高。

## 7.2 方案选型

选择方案最重要的依据就是数据量和时效性要求。

|  |  |  |
| --- | --- | --- |
| 时效性 | 数据量 | 分群方案 |
| 能接受隔天 | 无所谓 | HIVE宽表 |
| 即时产生 | 千万以下 | OLAP宽表(Elasticsearch,Tidb,Clickhouse…) |
| 即时产生 | 亿级 | Bitmap方式(Clickhouse，`) |

本次课程选择用Clickhouse实现Bitmap方式存储。

## 7.3 准备工作

参考《尚硅谷大数据之Clickhouse》

## 7.4 任务目标

把hive中标签宽表数据，写入至Clickhouse的宽表。

## 7.5 设计分析

1. 在clickhouse中建立对应的宽表。
2. 因为并不是hive表到hive表，所以 并不能够直接用insert select 解决

需要先通过把数据查询成为Dataframe ，再通过行动算子写入至Clickhouse的宽表。

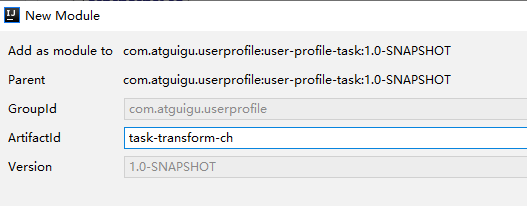
## 7.6 数据库准备

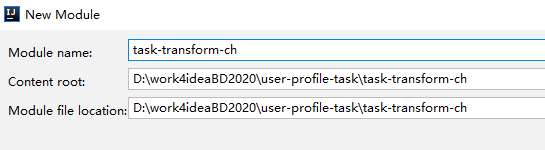
在clickhouse中建立数据

|  |
| --- |
| create database user\_profile2077; |

## 7.7 代码实现

### 7.7.1 搭建模块





### 7.7.2 在pom.xml中添加依赖及打包工具

|  |
| --- |
| <**dependencies**>   <**dependency**>  <**groupId**>com.atguigu.userprofile</**groupId**>  <**artifactId**>task-common</**artifactId**>  <**version**>1.0-SNAPSHOT</**version**>  </**dependency**> </**dependencies**>  <**build**>  <**plugins**>  *<!-- 该插件用于将Scala代码编译成class文件 -->* <**plugin**>  <**groupId**>net.alchim31.maven</**groupId**>  <**artifactId**>scala-maven-plugin</**artifactId**>  <**version**>3.4.6</**version**>  <**executions**>  <**execution**>  *<!-- 声明绑定到maven的compile阶段 -->* <**goals**>  <**goal**>compile</**goal**>  <**goal**>testCompile</**goal**>  </**goals**>  </**execution**>  </**executions**>  </**plugin**>   <**plugin**>  <**groupId**>org.apache.maven.plugins</**groupId**>  <**artifactId**>maven-assembly-plugin</**artifactId**>  <**version**>3.0.0</**version**>  <**configuration**>  <**descriptorRefs**>  <**descriptorRef**>jar-with-dependencies</**descriptorRef**>  </**descriptorRefs**>  </**configuration**>  <**executions**>  <**execution**>  <**id**>make-assembly</**id**>  <**phase**>package</**phase**>  <**goals**>  <**goal**>single</**goal**>  </**goals**>  </**execution**>  </**executions**>  </**plugin**>  </**plugins**> </**build**> |

### 7.7.3 添加配置文件

core-site.xml、hdfs-site.xml、hive-site.xml、hive-site.xml、log4j.properties与之前模块无异,主要用于本地调试。

config.properties

|  |
| --- |
| ***#mysql配置* mysql.url**=**jdbc:mysql://bigdata01:3306/user\_profile\_manager?characterEncoding=utf-8&useSSL=false mysql.username**=**root mysql.password**=**Flzx3qc!   clickhouse.url**=**jdbc:clickhouse://bigdata01:8123/user\_profile2077** |

添加执行clickhouse的Sql语句的工具类。

因为该工具也会被其他模块使用，所以放在task-

|  |
| --- |
| **object** MyClickHouseUtil {   **private val** *properties*: Properties = MyPropertiesUtil.*load*(**"config.properties"**)  **val** *CLICKHOUSE\_URL* = *properties*.getProperty(**"clickhouse.url"**)    **def** executeSql(sql: String ): Unit ={  Class.*forName*(**"ru.yandex.clickhouse.ClickHouseDriver"**);  **val** connection: Connection = DriverManager.*getConnection*(*CLICKHOUSE\_URL*, **null**, **null**)  **val** statement: Statement = connection.createStatement()  statement.execute(sql)  connection.close()   }  } |

### 7.7.4 主程序

|  |
| --- |
| **object** TaskTransformChApp {    **def** main(args: Array[String]): Unit = {   ***// 读取宽表数据* val** taskId: String = args(0)  **val** taskDate: String = args(1)  **val** sparkConf: SparkConf = **new** SparkConf().setAppName(**"task\_transform\_ch\_app"**)***//.setMaster("local[\*]")* val** sparkSession: SparkSession = SparkSession.*builder*().config(sparkConf).enableHiveSupport().getOrCreate()    **val** properties: Properties = MyPropertiesUtil.*load*(**"config.properties"**)   **val** userProfileDbName = properties.getProperty(**"user-profile.dbname"**)  **val** wareHouseDbName = properties.getProperty(**"data-warehouse.dbname"**)  **val** hdfsStorePath = properties.getProperty(**"hdfs-store.path"**)  **val** clickhouseUrl = properties.getProperty(**"clickhouse.url"**)  ***// 得到宽表的列数据* val** tableName=**"up\_tag\_merge\_"** + taskDate.replace(**"-"**,**""**)   ***//查询* val** tableDf: DataFrame = sparkSession.sql(**"select \* from user\_profile2021."**+tableName).toDF()   **val** columns: Array[String] = tableDf.columns   ***// 形成RDD* val** dropTableSql= **s"drop table IF EXISTS $**userProfileDbName**.$**tableName**"** MyClickHouseUtil.*executeSql*(dropTableSql,clickhouseUrl)  ***//建立clickhouse表  //create table user\_info  //(  // uid String,  // gender String,  // city String,  // date String  //)engine=MergeTree()  //order by (uid);* val** columnsSql: String = columns.map(column => column.toLowerCase+**" String"**).mkString(**","**)  **val** createTableSql= **"create table user\_profile2021."**+tableName + **"("**+columnsSql+**") engine =MergeTree() order by uid"** MyClickHouseUtil.*executeSql*(createTableSql,clickhouseUrl)   ***// 利用jdbc 算子写入clickhouse*** tableDf.write.mode(SaveMode.*Append*)  .option(**"batchsize"**, **"100"**)  .option(**"isolationLevel"**, **"NONE"**) ***// 关闭事务*** .option(**"numPartitions"**, **"4"**) ***// 设置并发*** .option(**"driver"**,**"ru.yandex.clickhouse.ClickHouseDriver"**)  .jdbc(clickhouseUrl,tableName,**new** Properties())   } } |

## 7.8 打包发布

同之前的任务

### 在流程任务中添加任务





因为要在合并宽表任务之后执行，所以级别设为300

调度任务之后



再次测试执行，等待结果。

## 7.9 验证结果

