为Spark 2.x添加ALTER TABLE ADD COLUMNS语法支持

Spark SQL从2.0开始已经不再支持ALTER TABLE table_name ADD COLUMNS (col_name data_type [COMMENT col_comment], ...)这种语法了（下文简称add columns语法）。如果你的Spark项目中用到了SparkSQL+Hive这种模式，从Spark1.x升级到2.x很有可能遇到这个问题。为了解决这个问题，我们一般有3种方案可以选择：

1、启动一个hiveserver2服务，通过jdbc直接调用hive，让hive执行add columns语句。这种应该是改起来最为方便的一种方式了，缺点就是，我们还需要在启动一个hiveserver服务，多一个服务依赖，会增加整个系统的维护成本。

2、SparkSQL+Hive这种模式，要求我们启动一个HiveMetastore服务，给SparkSQL用，我们也可以在代码中直接直接连接HiveMetastore去执行add columns语句。这种方式的好处是不需要额外依赖其他服务，缺点就是我们要自己调用HiveMetastore相关接口，自己管理SessionState，用起来比较麻烦。

3、最后一种方式就是直接修改Spark，让他支持add columns语法。这种方式最大的好处就是我们原有的业务逻辑代码不用动，问题就在于，要求对Spark源码有一定的了解，否则改起来还是挺费劲的。这也是我写这篇文章的目的：让大家能够参考本文自行为Spark添加add columns语法支持，下面的修改基于Spark 2.1.0版本。

为Spark添加add columns语法支持

改进语法定义

Spark2.1开始使用ANTLR来解析SQL语法，它的语法定义文件借鉴的Presto项目，我们在Spark源码中找到这个文件sql/catalyst/src/main/antlr4/org/apache/spark/sql/catalyst/parser/SqlBase.g4，做如下改动：

```
@@ -127,6 +127,8 @@ statement
       ('(' key=tablePropertyKey ')')?                    #showTblProperties
     | SHOW COLUMNS (FROM | IN) tableIdentifier
       ((FROM | IN) db=identifier)?                       #showColumns
+    | ALTER TABLE tableIdentifier ADD COLUMNS
+      ('(' columns=colTypeList ')')?                     #addColumns
     | SHOW PARTITIONS tableIdentifier partitionSpec?     #showPartitions
     | SHOW identifier? FUNCTIONS
       (LIKE? (qualifiedName | pattern=STRING))?          #showFunctions
@@ -191,7 +193,6 @@ unsupportedHiveNativeCommands
     | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=COMPACT
     | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=CONCATENATE
     | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=SET kw4=FILEFORMAT
-    | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=ADD kw4=COLUMNS
     | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=CHANGE kw4=COLUMN?
     | kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=REPLACE kw4=COLUMNS
     | kw1=START kw2=TRANSACTION
```

194行的kw1=ALTER kw2=TABLE tableIdentifier partitionSpec? kw3=ADD kw4=COLUMNS是在unsupportedHiveNativeCommands列表中，我们首先把它去掉。

为了让Spark能解析ALTER TABLE table_name ADD COLUMNS (col_name data_type [COMMENT col_comment], ...)，我们还需要在129行处新增| ALTER TABLE tableIdentifier ADD COLUMNS ('(' columns=colTypeList ')')? #addColumns最后的#addColumns是为了让ANTLR插件（这个插件定义在sql/catalyst/pom.xml中）为我们自动生成addColumns相关方法，便于我们做语法解析处理。这个语法中有2个参数需要我们处理table_name和columns。

改进SparkSqlAstBuilder，使其能处理addColumns

SparkSqlAstBuilder的作用是将ANTLR的语法树翻译为LogicalPlan/Expression/TableIdentifier

要修改的文件为：sql/core/src/main/scala/org/apache/spark/sql/execution/SparkSqlParser.scala，我们在178行处，新增如下方法：

```
override def visitAddColumns(ctx: AddColumnsContext): LogicalPlan = withOrigin(ctx) {
  val tableName = visitTableIdentifier(ctx.tableIdentifier())
  val dataCols = Option(ctx.columns).map(visitColTypeList).getOrElse(Nil)

  AlterTableAddColumnsCommand(tableName, dataCols)
}
```

visitAddColumns方法是ANTLR插件自动为我们生成的方法，定义在SparkSqlAstBuilder的父类AstBuilder中（AST，Abstract Syntax Tree
，抽象语法树），这个方法用来处理我们在SqlBase.g4中定义的| ALTER TABLE tableIdentifier ADD COLUMNS ('(' columns=colTypeList ')')? #addColumns，我们这里重载了visitAddColumns方法用来提取表名及新增的字段列表，并返回一个LogicalPlan：AlterTableAddColumnsCommand，这个类我们接下来会说明。

新增一个为表添加字段的命令

修改sql/core/src/main/scala/org/apache/spark/sql/execution/command/tables.scala，在120行处，新增AlterTableAddColumnsCommand类：

```
case class AlterTableAddColumnsCommand(
    tableName: TableIdentifier,
    newColumns: Seq[StructField]) extends RunnableCommand {

  override def run(sparkSession: SparkSession): Seq[Row] = {
    val catalog = sparkSession.sessionState.catalog
    val table = catalog.getTableMetadata(tableName)

    DDLUtils.verifyAlterTableType(catalog, table, isView = false)

    val newSchema = StructType(table.schema.fields ++ newColumns)
    val newTable = table.copy(schema = newSchema)
    catalog.alterTable(newTable)
    Seq.empty[Row]
  }
}
```

RunnableCommand类继承自LogicalPlan，run方法用于执行addColumns语法对应的执行逻辑。这个类的处理逻辑比较简单，就不详细介绍了。

修复HiveExternalCatalog无法修改表schema的问题

我们在第3步的AlterTableAddColumnsCommand中，虽然调用了catalog.alterTable(newTable)来修改表信息，但实际上并不能将新的字段添加到表中，因为Spark代码写死了，不能改Hive表的schema，我们还需要修改HiveExternalCatalog类（sql/hive/src/main/scala/org/apache/spark/sql/hive/HiveExternalCatalog.scala），改动如下：

```
@@ -588,7 +588,8 @@ private[spark] class HiveExternalCatalog(conf: SparkConf, hadoopConf:
Configurat
      val newTableProps = oldDataSourceProps ++ withStatsProps.properties + partitionProvide
```

```
rProp
      val newDef = withStatsProps.copy(
        storage = newStorage,
-       schema = oldTableDef.schema,
+       // allow `alter table xxx add columns(xx)`
+       schema = tableDefinition.schema,
        partitionColumnNames = oldTableDef.partitionColumnNames,
        bucketSpec = oldTableDef.bucketSpec,
        properties = newTableProps)
```

我们将591行的schema = oldTableDef.schema替换为schema = tableDefinition.schema即可。至此，我们完成了整个代码的调整。

最后参考Spark的编译文档： building Spark - Spark 2.1.0 Documentation ，将Spark编译打包即可。

Spark 2.x会将编译后的assembly放到jars目录下，我们这次的改动会影响到以下几个jar包：

spark-catalyst_2.11-2.1.0.jar
spark-sql_2.11-2.1.0.jar
spark-hive_2.11-2.1.0.jar

如果Spark已经部署过了，可以直接将以上3个jar替换掉。更新Spark后，我们就可以使用alter table xxx add columns(xx)了。

本文转自：自己动手为Spark 2.x添加ALTER TABLE ADD COLUMNS语法支持