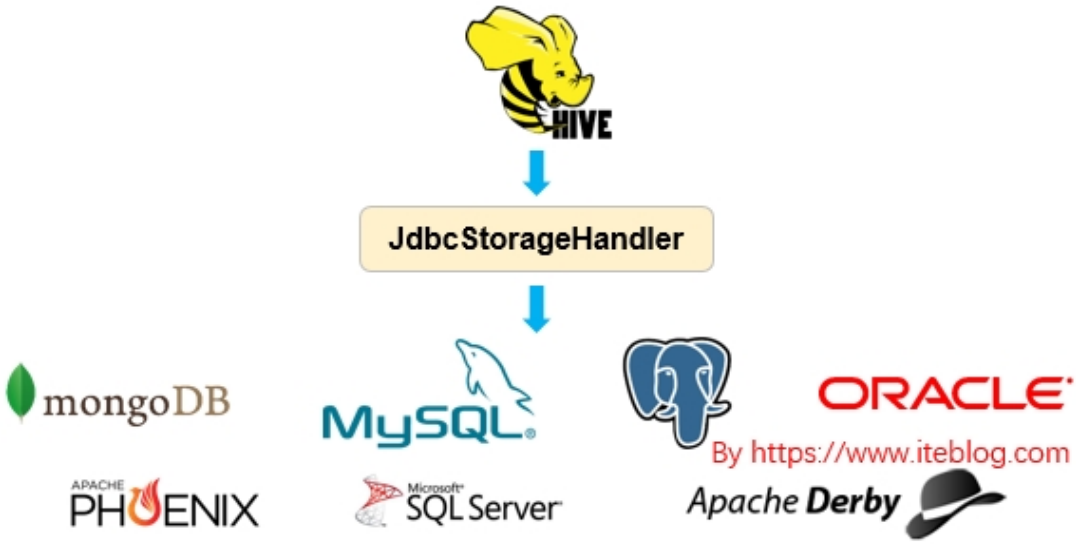
Apache Hive JdbcStorageHandler 编程入门指南

Apache Hive 从 HIVE-1555 开始引入了 JdbcStorageHandler ，这个使得 Hive 能够读取 JDBC 数据源，关于 Apache Hive 引入 JdbcStorageHandler 的背景可以参见 《Apache Hive 联邦查询（Query Federation）》。本文主要简单介绍 JdbcStorageHandler 的使用。



如果想及时了解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

语法

JdbcStorageHandler 使得 Hive 能够读取 JDBC 数据源，目前 JdbcStorageHandler 不支持将数据写入到 JDBC 数据源。为了使用 JdbcStorageHandler ，我们需要在 Hive 中创建外部表，具体如下：

```
CREATE EXTERNAL TABLE iteblog
(
  name string,
  age int,
  gpa double
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
    "hive.sql.database.type" = "MYSQL",
    "hive.sql.jdbc.driver" = "com.mysql.jdbc.Driver",
    "hive.sql.jdbc.url" = "jdbc:mysql://www.iteblog.com/sample",
    "hive.sql.dbcp.username" = "hive",
    "hive.sql.dbcp.password" = "hive",
```

```
    "hive.sql.table" = "STUDENT",
    "hive.sql.dbcp.maxActive" = "1"
);
```

我们可以使用 alter table 命令来修改表的 JdbcStorageHandler 属性，就和正常的表一样，如下：

```
ALTER TABLE iteblog SET TBLPROPERTIES ("hive.sql.dbcp.password" = "passwd");
```

JdbcStorageHandler 支持的表属性

必选属性

在 Hive 中使用 JdbcStorageHandler ，下面的属性是必须指定的

- hive.sql.database.type: JDBC 数据库类型，支持 MYSQL, POSTGRES, ORACLE, MSSQL, DERBY；
- hive.sql.jdbc.url: jdbc 链接字符串；
- hive.sql.jdbc.driver: jdbc driver 类；
- hive.sql.dbcp.username: jdbc 连接用户名；
- hive.sql.dbcp.password: jdbc 明文密码。强烈建议不要通过这个参数设置密码。推荐将密码存储在 keystore 中，详情参见下面的安全密码设置章节。
- hive.sql.table / hive.sql.query: 我们需要指定 "hive.sql.table" 或 "hive.sql.query" 来说明如何从 jdbc 数据库获取数据。 "hive.sql.table" 表示单个表，"hive.sql.query" 表示任意 sql 查询。

可选属性

除了上面的必选属性，JdbcStorageHandler 还支持以下几个可选属性：

- hive.sql.catalog: jdbc catalog 名字(仅仅在 hive.sql.table 被指定的时候才支持)
- hive.sql.schema: jdbc schema 名称 (仅仅在 hive.sql.table 被指定的时候才支持)
- hive.sql.jdbc.fetch.size: 每个批次获取的行数
- hive.sql.dbcp.xxx: 所有 dbcp 参数都将传递给 commons-dbcp。 详情请参见 https://commons.apache.org/proper/commons-dbcp/configuration.html。比如如果你在表的属性里面指定了 hive.sql.dbcp.maxActive=1 ， Hive 将会传递 maxActive=1 到 commons-dbcp。

支持的数据类型

JdbcStorageHandler 表中列支持的数据类型有：

- 数字数据类型： byte, short, int, long, float, double
- Decimal，支持 scale 和 precision
- String 数据类型： string, char, varchar
- Date
- Timestamp

复杂的数据类型，比如 struct, map, array 目前还不支持。

列和数据类型映射

hive.sql.table / hive.sql.query 使用模式定义表格数据，模式定义必须与表模式定义相同。
例如，以下 create table 语句将失败：

```
CREATE EXTERNAL TABLE iteblog
(
  name string,
  age int,
  gpa double
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
    . . . . . .
    "hive.sql.query" = "SELECT name, age, gpa, gender FROM STUDENT",
);
```

但是 hive.sql.table / hive.sql.query 模式的列名和列类型可能与表的模式不同。
在这种情况下，数据库列按位置映射到 hive 列；如果数据类型不同，Hive 将尝试根据 Hive 表模式转换它。 例如：

```
CREATE EXTERNAL TABLE iteblog
(
  sname string,
  age int,
  effective_gpa decimal(4,3)
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
    . . . . . .
```

```
    "hive.sql.query" = "SELECT name, age, gpa FROM STUDENT",
);
```

In case the conversion is not possible, Hive will produce null for the field.

Hive 将尝试将 STUDENT 表的 gpa 的 double 类型转换为 decimal(4,3) 作为 iteblog 表的 effective_gpa 字段。如果无法进行转换，Hive 将把该字段的值转换为 null。

Auto Shipping

如果在查询中使用了 JdbcStorageHandler，JdbcStorageHandler 会自动将所需的 jar 发送到 MR/Tez/LLAP 后端。用户无需手动添加 jar。 如果在 classpath 中检测到任何 jdbc 驱动程序的 jar（包括mysql、postgres、oracle 和 mssql），JdbcStorageHandler 还会将所需的 jdbc 驱动程序 jar 发送到后端。但是，用户仍然需要将 jdbc 驱动程序 jar 复制到 hive 的 classpath（通常是 hive 的 lib 目录）。

密码保护（Securing Password）

在大多数情况下，我们不希望在表属性“hive.sql.dbcp.password”中以明文的形式存储 jdbc 密码。相反，用户可以使用以下命令将密码存储在 HDFS 上的 Java 密钥库文件中：

```
hadoop credential create host1.password -provider jceks://hdfs/user/foo/test.jceks -v passwd1
hadoop credential create host2.password -provider jceks://hdfs/user/foo/test.jceks -v passwd2
```

这将在 hdfs://user/foo/test.jceks 里面创建一个 keystore 文件，其中包含两个密钥：host1.password 和 host2.password。在 Hive 中创建表时，我们需要在 create table 语句中指定“hive.sql.dbcp.password.keystore”和“hive.sql.dbcp.password.key”而不是“hive.sql.dbcp.password”，具体如下：

```
CREATE EXTERNAL TABLE iteblog
(
  name string,
  age int,
  gpa double
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
    . . . . . .
    "hive.sql.dbcp.password.keystore" = "jceks://hdfs/user/foo/test.jceks",
```

```
    "hive.sql.dbcp.password.key" = "host1.password",
    . . . . . .
);
```

我们需要通过仅授权目标用户读取此文件来保护 keystore 文件。Hive 将检查 keystore 文件的权限，以确保用户在创建/更改表时具有读取权限。

分区

Hive 能够拆分 jdbc 数据源并以并行的方式处理每个分片。用户可以使用以下表属性来决定是否拆分以及拆分的分片数：

- hive.sql.numPartitions: 为数据源生成多少个分片，如果不需要拆分则设置为 1
- hive.sql.partitionColumn: 需要对哪个列进行拆分。如果指定了这个，Hive 会将此列拆分成 hive.sql.numPartitions，每个分区的拆分点需要使用 hive.sql.lowerBound 和 hive.sql.upperBound 计算。如果没有指定这个参数，但 numPartitions > 1，Hive 将使用 offset 拆分数据源。但是，对于某些数据库，偏移量并不总是可靠的。如果要拆分数据源，强烈建议定义partitionColumn。partitionColumn 必须存在于 "hive.sql.table"/"hive.sql.query" 模式中。
- hive.sql.lowerBound / hive.sql.upperBound: 用于拆分 partitionColumn 计算间隔的下限/上限。两个属性都是可选的。如果未定义，Hive 将对数据源执行 MIN/MAX 查询以获得下限/上限。请注意，hive.sql.lowerBound 和 hive.sql.upperBound 都不能为 null。

使用示例如下：

```
TBLPROPERTIES (
    . . . . . .
    "hive.sql.table" = "DEMO",
    "hive.sql.partitionColumn" = "num",
    "hive.sql.numPartitions" = "3",
    "hive.sql.lowerBound" = "1",
    "hive.sql.upperBound" = "10",
    . . . . . .
);
```

这种表将会拆分成3个分片，num<4 or num is null, 4=7

```
TBLPROPERTIES (
    . . . . . .
    "hive.sql.query" = "SELECT name, age, gpa/5.0*100 AS percentage FROM STUDENT",
    "hive.sql.partitionColumn" = "percentage",
    "hive.sql.numPartitions" = "4",
    . . . . . .
);
```

Hive 将执行 jdbc 查询以获取 percentage 列的 MIN/MAX，这张表对应的 min/max 为 60/100。然后表将创建4个分片：(,70),[70,80),[80,90),[90,)。 第一个分片还包括空值。

如果要查看 JdbcStorageHandler 生成的分片，可以在 hiveserver2 日志或 Tez AM 日志中查找以下消息：

```
jdbc.JdbcInputFormat: Num input splits created 4
jdbc.JdbcInputFormat: split:interval:ikey[,70)
jdbc.JdbcInputFormat: split:interval:ikey[70,80)
jdbc.JdbcInputFormat: split:interval:ikey[80,90)
jdbc.JdbcInputFormat: split:interval:ikey[90,)
```

计算下推

Hive 会积极地将计算推送到 jdbc 表，因此我们可以充分利用 jdbc 数据源的计算能力。比如，我们有另外一张名为 iteblog_hadoop 表，如下：

```
CREATE EXTERNAL TABLE iteblog_hadoop
(
  name string,
  age int,
  registration string,
  contribution decimal(10,2)
)
STORED BY 'org.apache.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
    "hive.sql.database.type" = "MYSQL",
    "hive.sql.jdbc.driver" = "com.mysql.jdbc.Driver",
    "hive.sql.jdbc.url" = "jdbc:mysql://www.iteblog.com/sample",
    "hive.sql.dbcp.username" = "hive",
    "hive.sql.dbcp.password" = "hive",
    "hive.sql.table" = "VOTER"
```

);

那么下面的 Join 操作将会下推到 MySql 执行：

select * from iteblog join iteblog_hadoop on student_jdbc.name=voter_jdbc.name;

可以通过 explain 查看生成的执行计划

explain select * from iteblog join iteblog_hadoop on student_jdbc.name=voter_jdbc.name;

```
      . . . . . .
      TableScan
        alias: iteblog
        properties:
          hive.sql.query SELECT `t`.`name`, `t`.`age`, `t`.`gpa`, `t0`.`name` AS `name0`, `
t0`.`age` AS `age0`, `t0`.`registration`, `t0`.`contribution`
FROM (SELECT *
FROM `STUDENT`
WHERE `name` IS NOT NULL) AS `t`
INNER JOIN (SELECT *
FROM `VOTER`
WHERE `name` IS NOT NULL) AS `t0` ON `t`.`name` = `t0`.`name`
      . . . . . .
```

计算下推仅在 jdbc 表由 hive.sql.table 定义时才会发生。 Hive 将重写 hive.sql.query，并在 jdbc 表上进行更多计算。在上面的例子中，mysql 将运行查询并检索 join
的结果，而不是获取两个表的数据，然后在 Hive 中进行 join 操作。

目前支持算子下推的操作符包括 filter, transform, join, union, aggregation 以及 sort。

本文翻译自 JdbcStorageHandler