文章
Lele Yang · 七月 22, 2021 阅读大约需 6 分钟

FAQ 常见问题系列--Java 从IRIS数据库中读取Stream数据性能优化-Prefetch方式介绍

提示：本文包含在Java中通过JDBC Driver对Caché/IRIS数据库进行查询的示例代码。

近期有客户反应使用Java从老版本Caché中读取数据时，如果数据中包含long varchar, Caché数据库中与之对应的属性类型为%Stream.GlobalCharacter，即使实际上该流数据长度非常小，也会成十几倍的降低性能。

大家先来看一段代码,

```
public static void test99()
    {
        Statement stmt = null;
        ResultSet rs = null;

        int fetchSize = 100000;
        long before = System.currentTimeMillis();
        String sql="Select Title, Notes from My.Employee Where id=1";
        try {

            CacheDataSource ds = new CacheDataSource();

            ds.setURL("jdbc:Cache://123.123.123.1:1972/Samples");            //??Cach
é???
            ds.setUser("_SYSTEM");
            ds.setPassword("SYS");
            Connection connection = ds.getConnection();
            connection.setAutoCommit(false);
            stmt = connection.createStatement();
            rs = stmt.executeQuery(sql);
            long executed = System.currentTimeMillis();
            System.out.println("execute take miliseconds of:"+(executed-before));
            ResultSetMetaData rsmd = rs.getMetaData();
            int colnum = rsmd.getColumnCount();
            String str = null;
            while (rs.next()) {
                for (int i = 1,ilen = colnum; i <= ilen; i++) {
                str = rs.getString(i);
                }
            }
          stmt.close();
          rs.close();
            connection.close();
            long end = System.currentTimeMillis();
            System.out.println("read take miliseconds of:"+(end - executed));
            System.out.print("total take miliseconds of:"+(end-before));
        }catch (Exception ex) {
              System.out.println("TinyJDBC caught exception: "
                   + ex.getClass().getName() + ": " + ex.getMessage());
```

```
        }
    }
```

以上代码中查询的sql表My.Employee,在Caché数据中对应的类如下，

```
Class My.Employee Extends (%Persistent, %Populate)
{

Property Name As %String;

Property Title As %String;

Property Notes As %Stream.GlobalCharacter;
}
```

那么对于Stream数据的读取慢在哪儿，为什么会慢呢？执行完sql语句数据库服务器端返回的结果集中将包括Title这个String类型数据，但是并不包括Stream数据Notes，这么设计也是合理的，因为如果把一个几个M的流数据直接包含在结果集中返回，而应用程序并不需要使用这个流数据，那么这个代价花得就有点不值了。所以真正去获取这个流数据是在getString()时向数据库服务器端发送请求然后服务器端将这个数据返回的，即使流数据的实际长度很小，网络的一来一回也无法避免。所以慢在getString, 原因是多花费了一次网络的往返。

为了应对此场景，IRIS增加了Prefetch机制，也就是预先读取一定长度的流数据放在结果集中一并返回，如果设置得恰当就能避免二次请求。以此提高读取性能。Prefetch使用方法请参考如下示例代码，

```
public static void test98()
    {
        Statement stmt = null;
        ResultSet rs = null;

        long before = System.currentTimeMillis();
        String sql="Select Title, Notes from My.Employee Where id=1";
        try {

            IRISDataSource ds = new IRISDataSource();
            ds.setURL("jdbc:IRIS://123.123.123.2:1972/USER/jdbc-
prefetch2.log");  // iris??????????IRIS2021.1????
            ds.setUser("_SYSTEM");
            ds.setPassword("SYS");
            IRISConnection irisconnection = (IRISConnection)ds.getConnection();
            irisconnection.setStreamPrefetchSize(10000);
            stmt = irisconnection.createStatement();
            rs = stmt.executeQuery(sql);
            long executed = System.currentTimeMillis();
            System.out.println("execute take miliseconds of:"+(executed-before));
            ResultSetMetaData rsmd = rs.getMetaData();
            int colnum = rsmd.getColumnCount();
            String str = null;
            while (rs.next()) {
                for (int i = 1,ilen = colnum; i <= ilen; i++) {
                str = rs.getString(i);
                }
            }
          stmt.close();
          rs.close();
            irisconnection.close();
            long end = System.currentTimeMillis();
            System.out.println("read take miliseconds of:"+(end - executed));
```

```
            System.out.print("total take miliseconds of:"+(end-before));
        }catch (Exception ex) {
              System.out.println("TinyJDBC caught exception: "
                   + ex.getClass().getName() + ": " + ex.getMessage());
        }
    }
}
```

以上代码中查询的sql表My.Employee在IRIS中的定义同上。

关于如何使用Java的更多内容，请参见如下在线文档，
https://docs.intersystems.com/irislatest/csp/docbook/Doc.View.cls?KEY=BJAVA

IRISDataSource API，请参见如下在线文档，
https://docs.intersystems.com/irislatest/csp/docbook/Doc.View.cls?KEY=BJAVArefapi#BJAVArefapiiris-data-source

#Java #JDBC

源URL:
https://cn.community.intersystems.com/post/faq-%E5%B8%B8%E8%A7%81%E9%97%AE%E9%A2%98%E7%B3%BB%E5%88%97-java-%E4%BB%8Eiris%E6%95%B0%E6%8D%AE%E5%BA%93%E4%B8%AD%E8%AF%BB%E5%8F%96stream%E6%95%B0%E6%8D%AE%E6%80%A7%E8%83%BD%E4%BC%98%E5%8C%96-prefetch%E6%96%B9%E5%BC%8F%E4%BB%8B%E7%BB%8D