Normalize TiKV Java client for TiSpark

Project Details

Description

Currently, TiSpark maintains a com.pingcap.tikv/tikv-client module.

This is a fork of the previous tikv-client package. Some TiSpark-only enhancements are added gradually, and the fork differs from the current official tikv-client.

This presents some difficulties:

- Enhancements and bug fixes made for the TiSpark tikv-client will not affect the official tikv-client and vice versa.
- It's really hard for TiSpark developers to keep up with the fast TiKV development speed, especially now that TiKV is released every two months. This is a big development burden.

Presenting the user with two options is equally confusing and difficult to choose.

Instead, use upstream client-java. This will bring the following benefits.

- Easier to maintain
- Prepare for integration with TiBigdata

Expected Features

We want to normalize the Java client and use the official one. This includes :

1. Changed from TiSpark tikv-client to official tikv-client.

2. Test new clients for compatibility. All current behavior should be preserved.

3. For any incompatibilities, change the upstream tikv-client or TiSpark.

This link has a more detailed description click.

My possible solution

At present, my idea is to add a dependency on tikv-client-java to the tikv-client module of TiSpark, and then start with the core module of TiSpark, replace the interface calls of the functions for tikv-client, and then deal with the errors caused by the replacement one by one. to resolve.

If there are too many errors that cannot be solved one by one, a conversion compatibility layer can be made between the original function and the new dependency to deal with some different requirements. This compatibility conversion layer can be gradually simplified in the future. There are 3 ways to deal with incompatible parts.

1. First, the upstream client-java project raises a pull request or an issue, and requests a solution (for example, support for cluster index)

2. For the parts that cannot use the upstream, move the functions into the adaptation layer TiDB.

3. For incompatible conversion parts, also migrate to the adaptation layer TIDB.

The way to detect the completion of the goal is to run integration tests to ensure that the original function is not affected.

Refer to TiBigData for parts suitable for placement in the adaptation layer.

The following is the dependency table of some core modules in TiSpark for client-java modules.

class&interface	rely
TiBatchWrite	TiBatchWriteException
	TiSession
	TTLManager
	TiConfiguration
	StoreVersion
	TiDBJDBCClient

	allocator.RowIDAllocator
	codec.TableCodec
	BytePairWrapper
	key.{Handle, IndexKey, IntHandle, RowKey}
	meta.{TiColumnInfo, TiDBInfo, TiIndexInfo, TiTableInfo}
TwoPhaseCommitter	util.ConcreteBackOffer
	TwoPhaseCommitter
	BytePairWrapper
	ByteWrapper
WriteUtil	types.DataType
	row.ObjectRowImpl
	codec.{CodecDataOutput, TableCodec}
	ConvertOverflowException
	TiDBConvertException
	key.{CommonHandle, Handle, IndexKey, IntHandle, RowKey}

	meta.{TiIndexColumn, TiIndexInfo, TiTableInfo}
TiStrategy	IgnoreUnsupportedTypeException
	expression._
	meta.TiDAGRequest.PushDownType
	meta.{TiDAGRequest, TiTimestamp}
	predicates.{PredicateUtils, TiKVScanAnalyzer}
	region.TiStoreType
	statistics.TableStatistics
TiDB Table	TiSession
	TiBatchWriteException
	key.Handle
	meta.{TiDAGRequest, TiTableInfo, TiTimestamp}

TimeLine

time	works
May 18 ~ June 18	1. Replace the dependency on the tikv-client module in the core module. 2. deal with syntax errors and usage errors of API and method calls caused by using the new dependency (client-java).
June 18 ~ July 18	1. Compare the interface differences between the tikv-client module and the upstream client-java, replacing the implementation class. 2. Investigate the implementation differences, and separate them to form a conversion layer.
July 18 ~ August 17	1. Try reducing translation layer code, updating upstream, or migrating functionality to the adaptation layer.

About Me

Contact Information

- **Name: ZiJian Ren**
- **Major: Software Engineering**
- **Email: yegetables@gmail.com**
- **WeChat: yegetables**
- **Github: https://github.com/ajian2002**
- **Blog: https://blog.yegetables.cn**
- **TimeZone: Shanghai, China (GMT+8)**

Education & Background

I am a software engineering student at XUPT University and a member of the XiYou Linux Group. With more than two years of experience in using Linux, java, c language experience, good at database and java

web back-end development. Familiar with database usage and basic principles have a simple understanding of the Raft distributed consensus protocol, love code, and technology, spend 30+ hours a week learning technology, and enjoy the fun of coding.

Open Source Contributions

I'm passionate about open source and I'd love to write code for the open-source community. Here are some of my experiences with open-source projects.

Contribution	Project
Normalize TiKV Java client for TiSpark	#2284[WIP]
Implement the database built-in function PI()	#1821 #1893 matrixone
Database course experiment lab	tinyKV MIT-6.830
Personal blog web backend	Typecho-Java
Feedback build errors	TiDB & TiFlash #33266

Others

Are you applying for other Projects?

No.

Blogging about the project

Yes, I will record the problems encountered, solutions and read the relevant source code of TiSpark and TiKV when completing this project, and record it in my blog so that the community and mentors can continue to track my progress, or you can also You can share your own understanding of the project, get guidance from the community, or provide some information for latecomers.

Commitments

At the beginning of July, I need some time to prepare for the school final exam.

I have no other plans for the summer. I will dedicate most of my time to this project.

Epilogue

I love writing code, I am willing to contribute to the open-source community, and I am passionate about spending time and energy learning about technologies that interest me. For me, it is a great pleasure to exchange technical issues with like-minded people.

Since I have been in contact with the open-source community in the past few months, the community members have given me great help, and I hope to contribute to the community. And I am willing to continue to communicate with the community and contribute code after the GSoC is over.

My heartfelt thanks to the GSOC and CNCF communities and the community members of the TiSpark project!