Data Transmission Service
SDK Documentation
Product Introduction

Contents

SDK Documentation
    Data Subscription SDK

SDK Documentation
Data Subscription SDK

Last updated：2018-09-11 11:31:38

Downloading Data Subscription SDK

Click to Download

Logs of SDK Releases

Version 2.6.0

1. Supported subscribing to multiple channels via a single SDK
2. Supported subscribing to "stop", "start" and other operations on Client
3. Supported the serialization of DataMessage.Record
4. Optimized the SDK performance and reduced the resource consumption

Version 2.5.0

1. Fixed bugs occurred with small probability in high-concurrency scenarios

2. Supported the globally unique auto-increment ID recorded in transactions

Version 2.4.0

1. Optimized the subscription logic by working with the backend to accurately display SDK's current consumption time point

2. Fixed the problem occurred while encoding a few special characters at the backend.

3. **Fixed multiple compatibility issues. We recommend that users who use older versions upgrade the software to this version ASAP.**

Overview of Data Subscription SDK Sample Code

The sample code of Tencent Cloud Binlog subscription is as follows:

```
public class Main {

public static void main(String[] args) throws Exception {
//Create a context
SubscribeContext context=new SubscribeContext();

//User secretId, secretKey
context.setSecretId("AKID-522dabaa14dceed746ba8ccfb58e9e6f");
context.setSecretKey("AKEY-0ff4c4557c1183fc572baecfa505869d");

//Subscription serviceIp and servicePort
context.setServiceIp("10.108.112.24");
context.setServicePort(50120);

//Create a client
SubscribeClient client=new DefaultSubscribeClient(context);
//Create a subscription listener
ClusterListener listener= new ClusterListener() {
@Override
public void notify(List<ClusterMessage> messages) throws Exception {
//Consume subscribed data
for(ClusterMessage m:messages){
for(Record.Field f:m.getRecord().getFieldList()){
if(f.getFieldname().equals("id")){
System.out.println("seq:"+f.getValue());
}
}
//Confirm consumption
m.ackAsConsumed();
}
}
@Override
public void onException(Exception e){
System.out.println("listen exception"+e);
}};
//Add a listener
client.addClusterListener(listener);
//Set the subscription channel requested
client.askForGUID("dts-channel-B2eG8xbLvi472wV3");
//Launch the client
client.start();
}
}
```

The whole process is a intuitive, typical producer-consumer model. As a consumer, SDK constantly pulls subscribed Binlog data from the server, consumes data, and confirms data consumption.

1. First, configure parameters and create a consumer client `SubscribeClient` .
2. Next, create a listener `ClusterListener` to consume the received Binlog subscription data, and return a confirmation message after consumption.

3. Finally, launch the client to start the process.
   The listener `ClusterListener` allows you to operate on the data received based on your own needs and filter the received Binlog data by type, for example, filtering out all `drop` statements.

   In the sample code, you need to provide five parameters. Among them, `secretId` and `secretKey` are key values associated with your Tencent Cloud account, which can be viewed in **Tencent Cloud console** -> **Cloud Products** -> **Cloud API Key** -> **API Key**. SDK uses these two parameters to authenticate your operations. The other three parameters `serviceIp` `servicePort` `channelId` are related to your Binlog subscription, which will be displayed on the console after the subscription contents are configured on the relevant pages of Tencent Cloud TencentDB for MySQL. For more information, please see Console Operation Guide.

   Note: Data Subscription SDK has been connected to CAM. Root account has all the permissions by default, which can be accessed with cloud API key of the root account. Sub-account has no permission by default, which must be given the access to the operation `name/dts:AuthenticateSubscribeSDK` , or the access to all DTS operations `QcloudDTSFullAccess` by the root account.

SDK API Description

SubscribeContext Class

Class description

This is mainly used to set user's SDK configuration information, including security credential secretId, secretKey, IP and port of subscription service.

Construction method

public SubscribeContext()

Class method

Set security credential secretId

Function prototype

public void setSecretId(String secretId)

Input parameters

Parameter Name	Type	Description
secretId	String	Security credential secretId, which can be viewed in **Tencent Cloud Console** -> **Cloud Products** -> **Cloud API Key** -> **API Key**

Returned result

None

Thrown exception

None

Set security credential secretKey

Function prototype

public void setSecretKey(String secretKey)

Input parameters

Parameter Name	Type	Description
secretKey	String	Security credential secretKey, which can be viewed in **Tencent Cloud Console** -> **Cloud Products** -> **Cloud API Key** -> **API Key**

Returned result

None

Thrown exception

None

Set the subscription service IP address

Function prototype

public void setServiceIp(String serviceIp)

Input parameters

Parameter Name	Type	Description
serviceIp	String	IP address of subscription service, which can be viewed on the Subscription Channel Configuration page of the console

Returned result

None

Thrown exception

None

Set the subscription service port

Function prototype

public void setServicePort(String servicePort)

Input parameters

Parameter Name	Type	Description
servicePort	String	Port number of subscription service, which can be viewed on the Subscription Channel Configuration page of the console

Returned result

None

Thrown exception

None

API SubscribeClient and API DefaultSubscribeClient

The `DefaultSubscribeClient` class implements the API `SubscribeClient`.

Class description

This is used to build the client program for subscription SDK, i.e. consumer for Binlog messages.

Based on user requirements, `DefaultSubscribeClient` provides two implementation methods: sync confirmation and async confirmation. In sync mode, a confirmation message is synchronously received each time the client consumes a Binlog message, to ensure that message consumption confirmation can be received by the server as soon as possible. In this mode, the overall performance of SDK is lower compared to async mode. In async mode, the consumer program confirms message consumption asynchronously, that is, message pulling and confirmation are processed asynchronously and independently, in which case, the performance is higher than that in sync confirmation mode. Users may select a confirmation mode as needed.

Construction method

Construct DefaultSubscribeClient

Function prototype

public DefaultSubscribeClient(SubscribeContext context, boolean isSync) throws Exception

Input parameters

Parameter Name	Type	Description
context	SubscribeContext	Configuration information of user SDK
isSynce	boolean	Whether sync consumption mode is used for SDK

Returned result

`DefaultSubscribeClient` instance

Thrown exception

- IllegalArgumentException: This exception is thrown when any parameter is invalid in the parameter context submitted by a user. Invalid situations: no security credential or incorrect format; no service IP/port or incorrect format.
- Excetion: This exception is thrown when an internal error occurred while initializing the SDK.

Construct DefaultSubscribeClient

Function prototype

public DefaultSubscribeClient(SubscribeContext context) throws Exception

Input parameters

Parameter Name	Type	Description
context	SubscribeContext	Configuration information of user SDK

Returned result

`DefaultSubscribeClient` instance. Default is async confirmation.

Thrown exception

- IllegalArgumentException: This exception is thrown when any parameter is invalid in the parameter context submitted by a user. Invalid situations: no security credential or incorrect format; no service IP/port or incorrect format.
- Excetion: This exception is thrown when an internal error occurred while initializing the SDK.

Class method

Add a listener for SDK consumer client

Function description

Add the listener `ClusterListener` to a `SubscribeClient` before subscribing to the incremental data in the channel.

Function prototype

public void addClusterListener(ClusterListener listener) throws Exception

Input parameters

Parameter Name	Type	Description
listener	ClusterListener	Listener to be used by a consumer client. The main process to consume Binlog messages should be implemented in `ClusterListener`

Returned result

None

Thrown exception

- IllegalArgumentException: This exception is thrown if the listener parameter submitted by user is empty.
- Exception: SDK only supports one listener. This exception is thrown when several listeners are added.

Request incremental data in a subscription channel

Function prototype

public void askForGUID(String channelId)

Input parameters

Parameter Name	Type	Description
channelId	String	Subscription channel ID, which can be viewed on the Subscription Channel Configuration page of the console

Returned result

None

Thrown exception

None

Launch SDK client

Function prototype

public void start() throws Exception

Input parameters

None

Returned result

None

Thrown exception

- Exception: This exception is thrown if an internal error occurred while launching the SDK.

Stop SDK client

Function prototype

public void stop(int waitSeconds) throws Exception

public void stop() throws Exception

Input parameters

Parameter Name	Type	Description
waitSeconds	int	Waiting time (in sec), which indicates how long does it take to forcedly stop SDK operation

"stop" function with no parameters will wait for a period of time for the thread to stop, which may last longer and be subject to the system scheduling. It is recommended to use the stop function with timeout for scenarios where specific restart time is required.

Returned result

None

Thrown exception

- Exception: This exception is thrown if an internal error occurred while stopping the SDK.

API ClusterListener

API Description

This is a callback API. An SDK user should implement the notify function of this API to consume subscription data, and handle exceptions that may occur during the consumption process by implementing the onException function.

API function

Notify SDK consumer client of subscription messages

Function description

This is mainly used to implement the consumption of incremental data. However, SDK will notify ClusterListener of the subscription data via the notify function when it receives the data.

Function prototype

public abstract void notify(List messages) throws Exception

Input parameters

Parameter Name	Type	Description
messages	List	Subscription data array. For more information on how to implement ClusterMessage, please see its definition.

Returned result

None

Thrown exception

Any exception during the consumption of subscription data will be thrown to the onException function implemented by users who will then handle these exceptions as needed.

Handle exceptions occurred while consuming subscription data

Function description

This is mainly used to handle exceptions occurred while consuming subscription data. Users can implement their own secure exit policy in onException.

Function prototype

public abstract void onException(Exception exception)

Input parameters

Parameter Name	Type	Description

Parameter Name	Type	Description
exception	Exception	Exception class in Java standard library

Returned result

None

Thrown exception

None

ClusterMessage Class

Class Description

The ClusterMessage class delivers consumed subscription data through the notify function. Each ClusterMessage saves data records of one **transaction** in TencentDB for MySQL, and each record in the transaction is saved via Record.

Class method

Obtain records from ClusterMessage

Function prototype

public Record getRecord()

Input parameters

None

Returned result

Type	Parameter Description
Record	Change record corresponding to a specific record in a transaction, such as begin, commit, update, insert, etc.

Thrown exception

None

Confirm consumed data

Function description

This is used to send the confirmation of consumed data to the subscription server. This function executes sync or async confirmation according to the value configured in SubscribeClient. Users must call this data after consuming data. Otherwise, normal logic may be affected, in which case the SDK may receive duplicate data.

Function prototype

public void ackAsConsumed() throws Exception

Input parameters

None

Returned result

None

Thrown exception

- Exception: This exception is thrown if an internal error occurred during the confirmation process.

Record Class

Class description

This indicates a certain record in subscribed Binlog data, generally, a member of a certain transaction `ClusterMessage` . The record may be a begin, commit or update statement.

Class method

Obtain the attribute value of Record

Function prototype

public String getAttribute(String key)

Input parameters

Parameter Name	Type	Description
key	String	Name of attribute value

Possible attribute key values are:

Attribute Key Value	Description
record_id	Record ID, which is automatically added by string in sequence to the channel, but cannot be ensured to be added continuously
source_type	Engine type of the database instance of Record. Available value: mysql
source_category	Record type. Available value: full_recorded
timestamp	The time when the Record is stored into binlog. This is also the time when the SQL statement is executed in TencentDB
checkpoint	File check point of Record, in the format of file_offset@file_name. "filen_name" is the number suffix of the binlog file
record_type	Operation type of Record. Available values: insert/update/delete/replace/ddl/begin/commit/heartbeat
db	Database name of the Record update table
table_name	Name of Record update table
record_encoding	Encoding of Record
primary	Name of the primary key column of Record update table
fields_enc	Encoding of each field value of Record. Fields are separated by commas, and empty value is used for non-character type

Returned result

Type	Parameter Description
String	Attribute Value

Thrown exception

None

Obtain change type of a record

Function prototype

public DataMessage.Record.Type getOpt()

Input parameters

None

Returned result

Type	Parameter Description
DataMessage.Record.Type	Record type

Possible values for DataMessage.Record.Type: insert, delete, update, replace, ddl, begin, commit and heartbeat. "heartbeat" is a heartbeat table internally defined for data transfer and mainly used to check health status of a subscription channel. Theoretically, a heartbeat is generated every second.

Thrown exception

None

Obtain Checkpoint of a record in Binlog

Function prototype

public String getCheckpoint()

Input parameters

None

Returned result

Type	Parameter Description
String	Checkpoint of a record in Binlog, in the format of binlog_offset@binlog_fid. "binlog_offset" is the offset of the change record in binlog file, and "binlog_fid" is the name of binlog file.

Thrown exception

None

Obtain the timestamp of a record in Binlog

Function prototype

public String getTimestamp()

Input parameters

None

Returned result

Type	Parameter Description
String	Timestamp string

Thrown exception

None

Obtain the database name of record

Function prototype

public String getDbname()

Input parameters

None

Returned result

Type	Parameter Description
String	Database name string

Thrown exception

None

Obtain the data table name of record

Function prototype

public String getTableName()

Input parameters

None

Returned result

Type	Parameter Description
String	Data table name string

Thrown exception

None

Obtain the primary key column name of record

Function prototype

public String getPrimaryKeys()

Input parameters

None

Returned result

Type	Parameter Description
String	Primary key column name. For composite primary keys, the column names are separated by commas

Thrown exception

None

Obtain database type of subscription instance

Function prototype

public DBType getDbType()

Input parameters

None

Returned result

Type	Parameter Description
DBType	Only TencentDB for MySQL is supported for data transfer, that is, DBType.MYSQL

Thrown exception

None

Obtain the number of fields in Record

Function prototype

public int getFieldCount()

Input parameters

None

Returned result

Type	Parameter Description
int	Number of fields in Record

Thrown exception

None

Check if Record is the first one in transaction

Function prototype

public Boolean isFirstInLogevent()

Input parameters

None

Returned result

Type	Parameter Description
Boolean	True: it is the first log in the transaction. False: it is not the first log

Thrown exception

None

Obtain the field definition list of a record table

Function prototype

public List getFieldList()

Input parameters

None

Returned result

Type	Parameter Description
List	Field array. For more information, please see the definition of Field class

Thrown exception

None

Field Class

Class description

The Field class defines the attributes of a field such as encoding, type, name, value, and whether it is a primary key.

Class method

Obtain the encoding format of a field

Function prototype

public String getFieldEnc()

Input parameters

None

Returned result

Type	Parameter Description
String	Field encoding of String type

Thrown exception

None

Obtain field name

Function prototype

public String getFieldname()

Input parameters

None

Returned result

Type	Parameter Description
String	Field name of String type

Thrown exception

None

Obtain data type of a field

Function prototype

public Field.Type getType()

Input parameters

None

Returned result

Type	Parameter Description
Field.Type	Field.Type is an enumeration type which corresponds to data types supported by MySQL, including INT8, INT16, INT24, INT32, INT64, DECIMAL, FLOAT, DOUBLE, NULL, TIMESTAMP, DATE, TIME, DATETIME, YEAR, BIT, ENUM, SET, BLOB, GEOMETRY, STRING, UNKOWN

Thrown exception

None

Obtain field value

Function prototype

public ByteString getFieldname()

Input parameters

None

Returned result

Type	Parameter Description
ByteString	Field value. It is NULL if left empty

Thrown exception

None

Check if the field is a primary key

Function prototype

public Boolean isPrimary()

Input parameters

None

Returned result

Type	Parameter Description
Boolean	True: the field is a primary key. False: it is not a primary key

Thrown exception

None