TDMQ for CMQ

Operation Guide

Product Documentation

Contents

Operation Guide
Queue Service
Queue Management

Last updated：2024-01-03 10:17:36

Overview

This document describes how to create a queue service and send a message in the TDMQ for CMQ console.

Entering the queue list

Log in to the TDMQ console, select **Queue Service** on the left sidebar, and select a region.

Creating a queue

1. On the queue list page, click **Create** and configure the basic information of the queue.

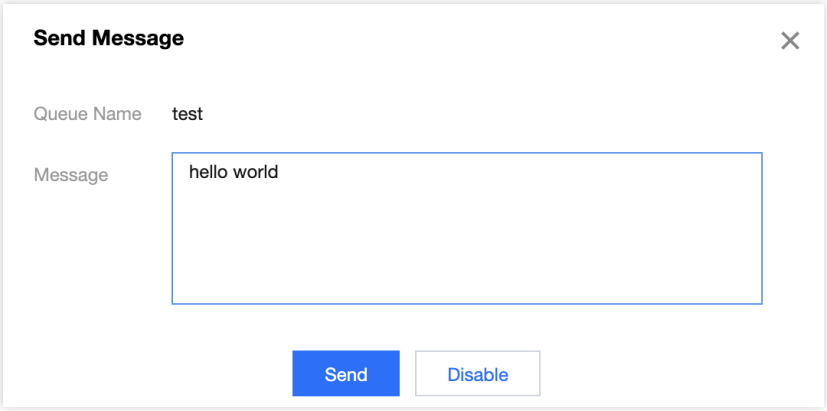
Parameter	Description
Queue Name	Queue name is case-insensitive and cannot be modified once created. It is the primary key of a queue service and the unique identifier of a resource. It is used to specify a queue when APIs are called for operations.
Resource Tag	It is optional and can help you easily categorize and manage TDMQ for CMQ resources in many dimensions. For detailed usage, see Managing Resource with Tag.
Max Message Unack Time	It ranges from 30 seconds to 12 hours. If the consumer client fails to acknowledge a received message within this time period, the server will automatically acknowledge the message. Even if you have set the delay time for a message, the max message unack time will still start from the time when the message is actually sent. For example, if the max message unack time is set to one hour while the message is set to be sent two hours later, the message will be deleted one hour later, causing message loss. Therefore, make sure the max message unack time is longer than the delay time of the message.
Long Polling Wait Time for Message Receipt	During the long polling wait time, a message consumption request will return a response only after a valid message is fetched or the long polling times out, which is similar to the

	long polling in Ajax requests. Value range: 0-30 seconds. A value lower than 3 seconds is recommended, as a high value may cause more duplicated messages.
Hidden Duration of Fetched Messages	A queue's `VisibilityTimeout` attribute in seconds. Value range: 1–43,200 seconds (1s–12 hours). A default value of `VisibilityTimeout` is set for each message. The visibility timeout duration starts when the worker receives the message. If the worker fails to process the message within this duration, the message may be received and processed by another worker.
Dead Letter Queue	A dead letter queue is used to process messages that cannot be consumed normally. After retry limit is reached, if the messages still cannot be consumed, it indicates that the consumer cannot properly consume the messages under normal circumstances. At this time, instead of immediately discarding the messages, MQ will send them to the special queue corresponding to the consumer.

2. Click **Next** to configure message rewind.
If the "Message Rewind" option is not enabled, a message that has been consumed by a consumer and confirmed for deletion will be deleted immediately.
If this option is enabled, you need to specify a "rewindable time range" not greater than the message lifecycle, preferably the same for easier troubleshooting.
3. Click **OK**, and you can see the created queue service in the queue service list.

Sending Messages

1. On the queue list page, click **Send Message** in the **Operation** column of the target queue.
2. Enter the message content and click **Send** to send a testing message to the recipient.



Note:
Enter the content to be sent of at least 1 byte. The maximum length is subject to the set `MaxMsgSize` attribute.

Resetting consumption status

After the consumption status is reset, heaped messages can be quickly redistributed to downstream consumers. In this way, the reset operation can solve the message heap problem without message loss.

Note:

On the queue list page, click **Reset Consumption Status** in the **Operation** column of the target queue, confirm your operation in the pop-up window, and the consumption status can be reset.

Viewing Queue Details

Last updated：2024-01-03 10:17:36

Overview

This document describes how to view the details of a created queue in the TDMQ for CMQ console.

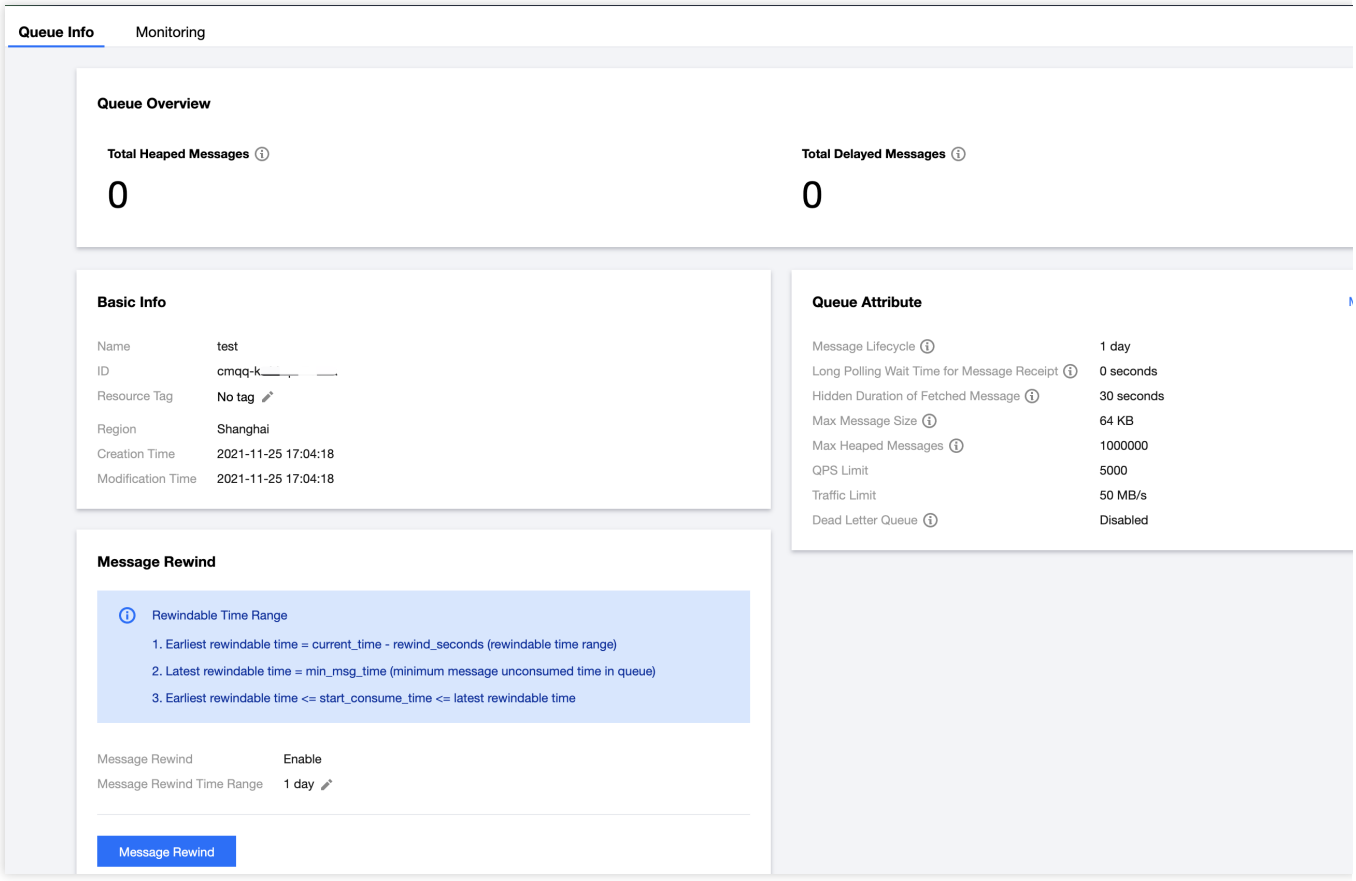
Directions

1. Log in to the TDMQ for CMQ console.
2. On the queue list page, click the ID of the target queue to enter the queue details page, where you can query:
Queue overview (shared cluster)
Visible Messages: A message is visible (in "Active" status) at first when it is sent to a queue, and it can be consumed by the consumer in this status.
Invisible Messages: A message is made invisible ("Inactive") after being fetched by the consumer, but it will become visible ("Active") again if it remains unconsumed after the `VisibilityTimeout` .
Capacity for Invisible Messages: Messages become invisible because they have not been acknowledged by the client timely. As invisible messages occupy certain memory space, each queue has a limit on the storage capacity for them. To increase the capacity, submit a ticket.
Delayed Messages: If there is no consumer online, the total number of delayed messages will be displayed as 0. After the consumer goes online, the real statistics will be displayed.
Capacity for Heaped Messages: Messages will be heaped if they are produced faster than consumed or if their consumption is blocked. Heaped messages occupy certain disk space, each queue has a limit on the storage capacity for them. To increase the capacity, submit a ticket.
API Calls in Last 24 Hours
Basic Information: Queue name, ID, resource tag, region, creation time, and modification time.
Queue Attributes: You can click **Modify Configuration** in the top-right corner to modify the queue attributes.

Parameter	Description
Max Message Unack Time	It ranges from 30 seconds to 12 hours. If the consumer client fails to acknowledge a received message within this time period, the server will automatically acknowledge the message. **Note: If you use delayed messages, make sure the delay time of the messages is shorter than the max message unack time. For example, if the max message unack time is set to one hour while the message is set to be sent two hours later, the message will be deleted one hour later.**
Long Polling	During the long polling wait time, a message consumption request will return a response only

Wait Time for Message Receipt	after a valid message is fetched or the long polling times out, which is similar to the long polling in Ajax requests. Value range: 0-30 seconds. A value lower than 3 seconds is recommended, as a high value may cause more duplicated messages.
Hidden Duration of Fetched Messages	During the long polling wait time, a message consumption request will return a response only after a valid message is fetched or the long polling times out, which is similar to the long polling in Ajax requests. Value range: 0-30 seconds. A value lower than 3 seconds is recommended, as a high value may cause more duplicated messages.
Max Message Length	A queue's `MaxMsgSize` attribute, which specifies the max size of the message body that can be sent to the queue. Unit: KB.
Max Size of Heaped Messages	Messages will be heaped if they are produced faster than consumed or if their consumption is blocked. Heaped messages occupy certain disk space, and the storage capacity for them is subject to that of the entire exclusive cluster. To increase the capacity, submit a ticket.
Max Message TPS	The frequency limit of calling the same API
Traffic Limit	The maximum throughput bandwidth of message production. If this limit is exceeded, the traffic will be throttled (the response time of production requests will increase).
Dead Letter Queue	A dead letter queue is used to process messages that cannot be consumed normally. After retry limit is reached, if the messages still cannot be consumed, it indicates that the consumer cannot properly consume the messages under normal circumstances. At this time, instead of immediately discarding the messages, MQ will send them to the special queue corresponding to the consumer.

Message Rewind: You can use this feature to consume a message again after it is successfully consumed and deleted by your business. For more information, see Message Rewind.



Viewing Monitoring Data

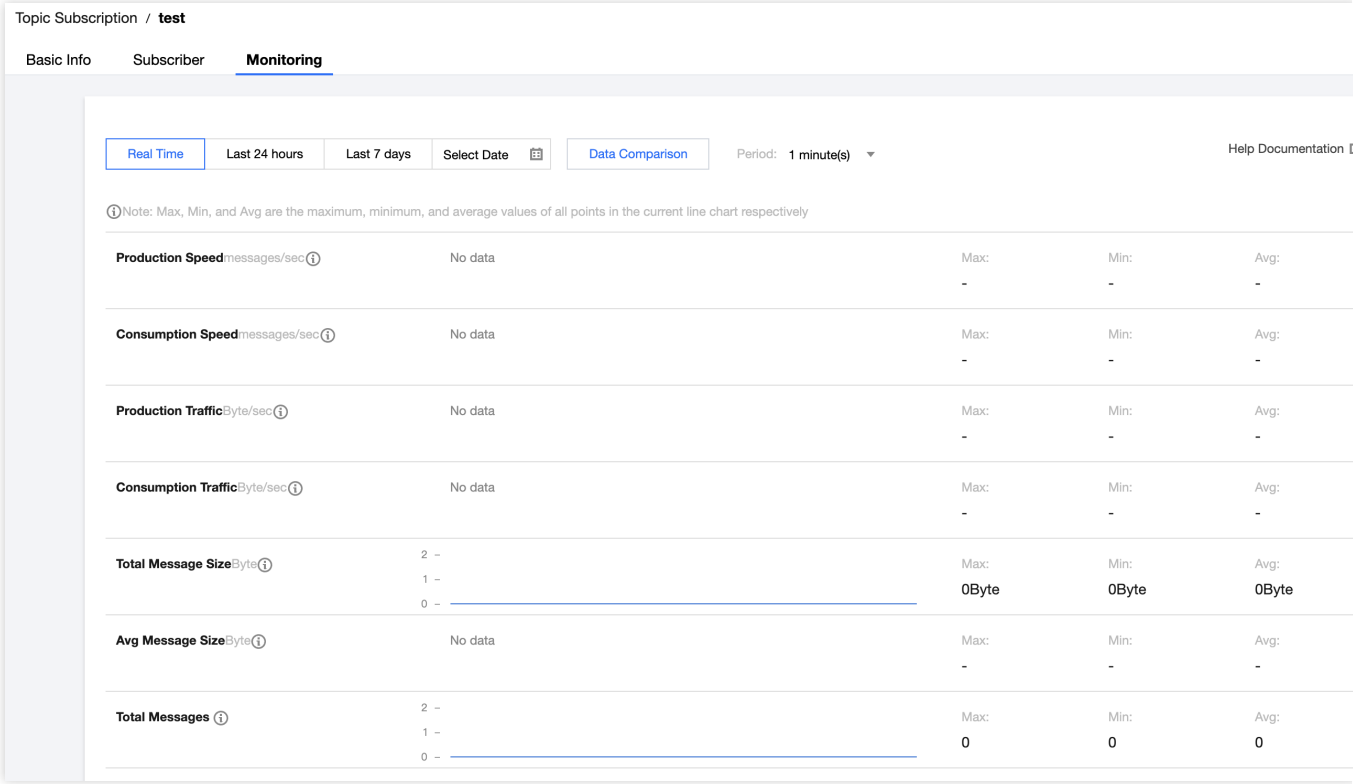
Last updated：2024-01-03 10:17:36

Overview

This document describes how to view the monitoring data of a created queue in the TDMQ for CMQ console.

Directions

1. Log in to the TDMQ for CMQ console.
2. Select **Queue Service** on the left sidebar, select a **Region**, and click the ID of the target queue to enter the queue details page.
3. On the queue details page, select the **Monitoring** tab at the top and select a time range to view the monitoring data of the queue.



Monitoring metric description

Monitoring Metric	Description

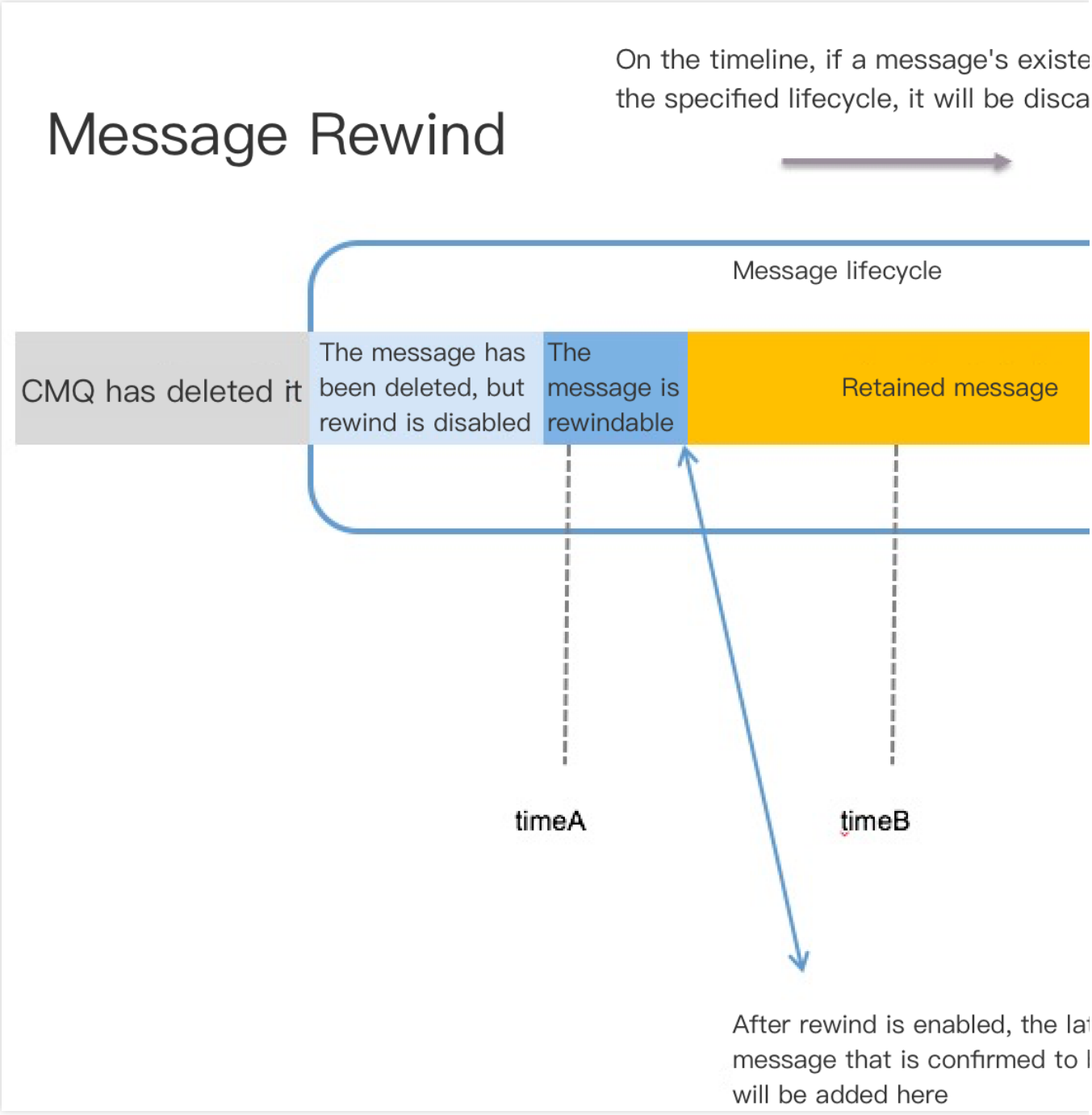
Heaped messages	`Activemessages` , which indicates the total number of messages that are in “Active” status in the queue in the selected time range. The metric value is approximate.
Invisible messages	A message will be made invisible (in “Inactive” status) after being fetched by the consumer. It will become visible (in “Active” status) again if it is not consumed after the `VisibilityTimeout` .
Production rate (messages/sec)	The number of messages produced by all producers in the queue per second in the selected time range.
Production rate (messages/sec)	The number of messages consumed by all consumers in the queue per second in the selected time range.
Production traffic (bytes/sec)	The size of messages produced by all producers in the queue per second in the selected time range.
Consumption traffic (bytes/sec)	The size of messages consumed by all consumers in the queue per second in the selected time range.
Total message size (bytes)	The total data size of current messages
Average message size (bytes)	The average data size of current messages
Total number of messages	The total number of current messages

Message Rewind

Last updated：2024-01-03 10:17:36

TDMQ for CMQ provides a message rewind feature similar to that in Kafka. After your business successfully consumes and deletes a message, you can use this feature to consume the message again.
This document describes the concepts, use cases, and use methods of message rewind in TDMQ for CMQ.

Feature



As shown above, the message lifecycle is circled in the blue box. After message rewind is enabled, messages consumed and deleted by consumers will be moved to the **rewindable message** section and retained on the TDMQ for CMQ backend. However, if the message existence exceeds the message lifecycle of the queue (assumed as 1 day), the message will be automatically deleted and cannot be rewound.

Product logic

Enable: if message rewind is not enabled, after a message is consumed by a consumer and its deletion is confirmed, it will be deleted immediately. When enabling this feature, you need to specify the rewind time range, which must be equal to or shorter than the message lifecycle.

Milestone: according to the policy above, after message rewind is enabled, the number of rewindable messages will keep increasing as consumers continuously consume and delete messages.
Disable: after message rewind is disabled, messages in the rewindable message section will be deleted immediately and cannot be rewound.
Queue attribute: message rewind is an attribute of a queue and can be set when you create the queue or modify its configuration. After specifying a rewind time point, all consumers will consume messages produced after this time point.
Billing: after message rewind is enabled, rewindable messages will incur certain retention fees. The unit price is calculated as part of message retention fees.
Specify rewind time point: when a consumer initiates rewind consumption, the queue name and specific rewind time need to be specified, and messages will be rewound from the maximum time point. The time is a `key` , and reverse consumption is not supported. You can consume from timeA to timeB/timeC but not vice versa as shown below.
Specify rewind time range: it ranges from 0 to 15 days. Only after message rewind is enabled in the console can deleted messages be rewound. You are recommended to always enable this feature for key applications and set the message rewind time range to the same as the message lifecycle.
Unable to specify message rewind for retained messages: if a message is retained and not consumed, you cannot specify a specific position for its consumption.

Rewindable range

The message rewindable range is sorted by message production time and is irrelevant to the order of deletion.
1. The maximum rewindable time point = current_time - rewind_seconds (retention period of rewindable message)
2. The minimum rewindable time point = min_msg_time (the minimum unconsumed message time of the queue)
3. Maximum rewindable time point ≤ start_consume_time ≤ minimum rewindable time point

Use Cases

This feature facilitates operations such as reconciliation and business system retry for core finance businesses.

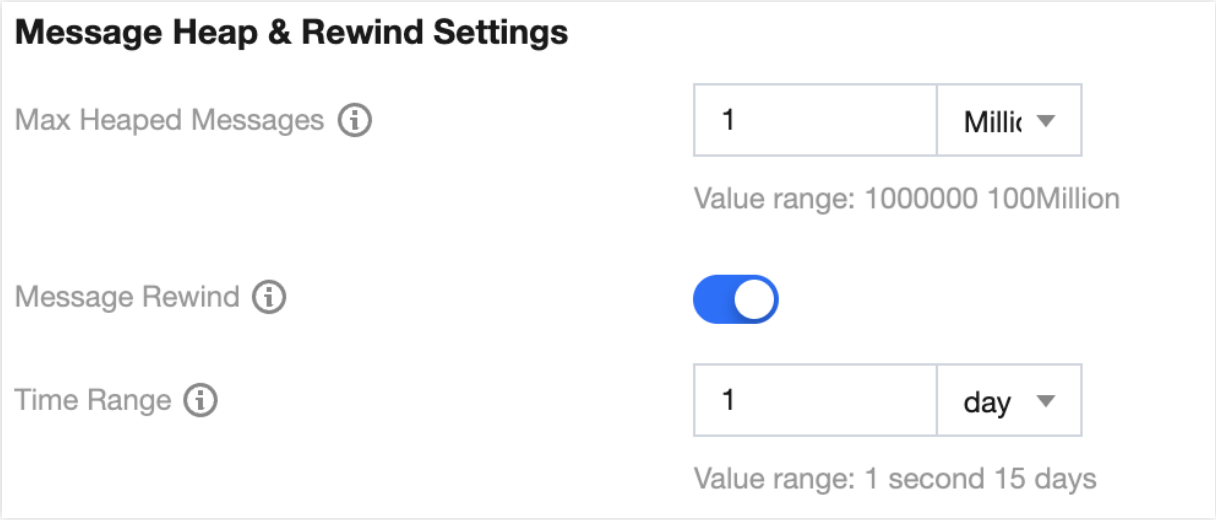
Use Case

The following is a typical use case of message rewind:
There are two businesses (A and B) in a normal production/consumption scenario. A produces messages and delivers them to a queue, and B consumes messages from the queue. At this point, A and B are decoupled from each other and don't care about each other. A only needs to produce and deliver messages, while B gets messages from the queue, deletes them from the queue, and then consumes them locally.

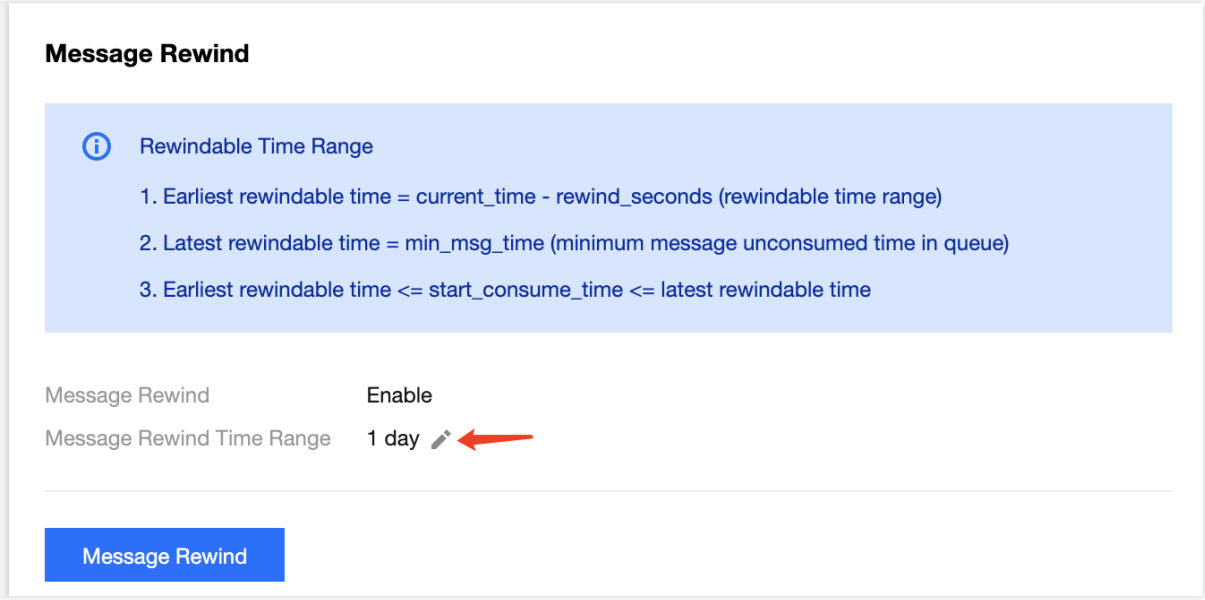
An exception occurs; for example, although business B tries to consume messages, consumption has been exceptional for a period of time. In this case, the messages have been deleted and cannot be consumed again, which will affect the business; moreover, it is necessary to suspend business B and wait for the development and OPS personnel to fix the exception before making business B online again. In addition, the personnel cannot monitor the status of business B in real time, so the exception may have already lasted some time before it is discovered.
To prevent this, business A should be concerned about the processing of business B, back up the produced messages, and make sure that business B can consume the message normally in the production environment before deleting the backup.
In this case, you can use the **message rewind** feature. After business B is restored, messages can be rewound to the last time point when business B's consumption was normal. Then, the messages obtained by business B will start at the specified time point, so business A doesn't need to care about any exceptions of business B at all. Business B should ensure the consumption idempotency.

Enabling message rewind

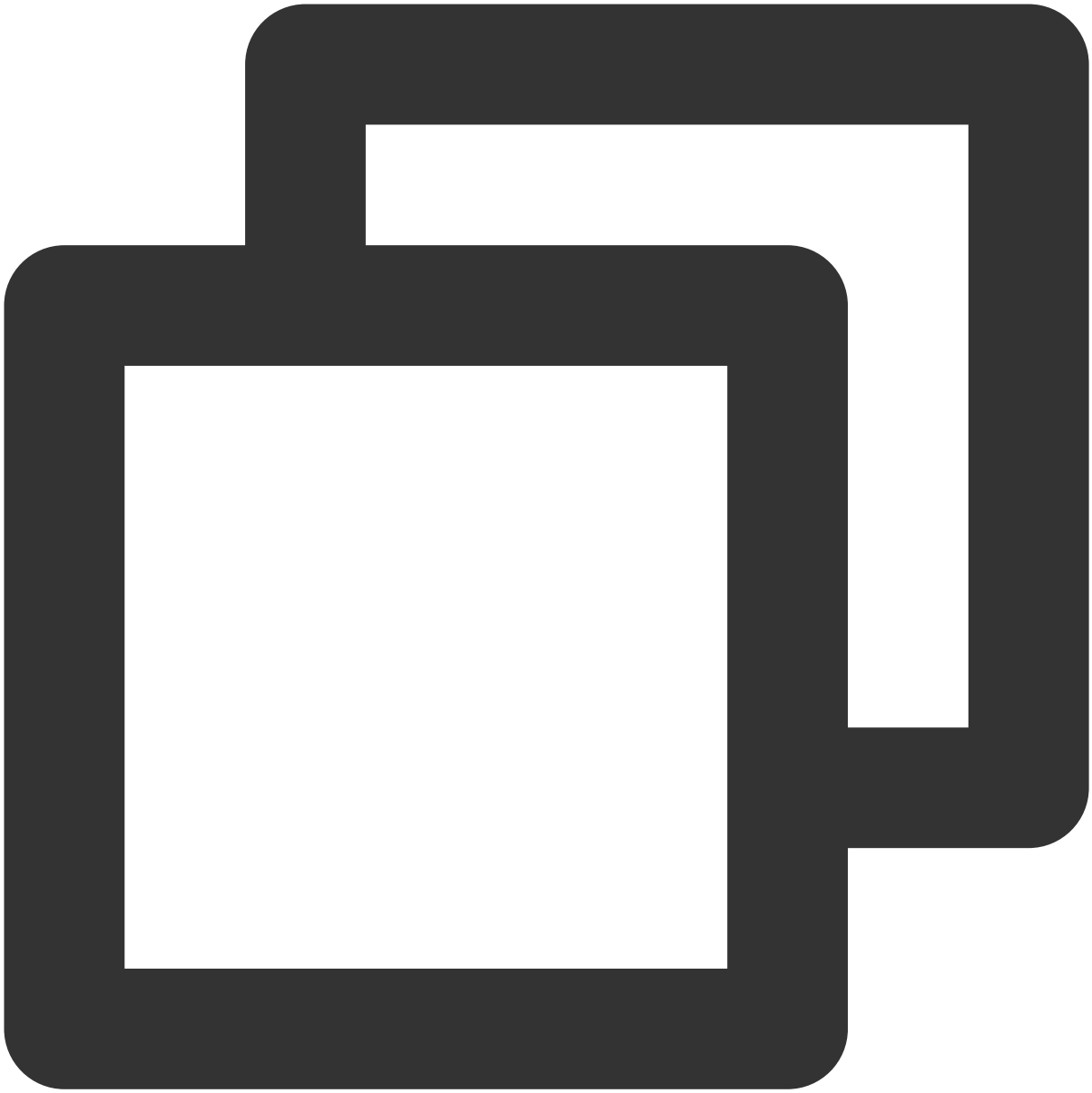
You can directly enable the message rewind feature when creating a queue in the TDMQ for CMQ console.



You can also enable message rewind on the queue details page.

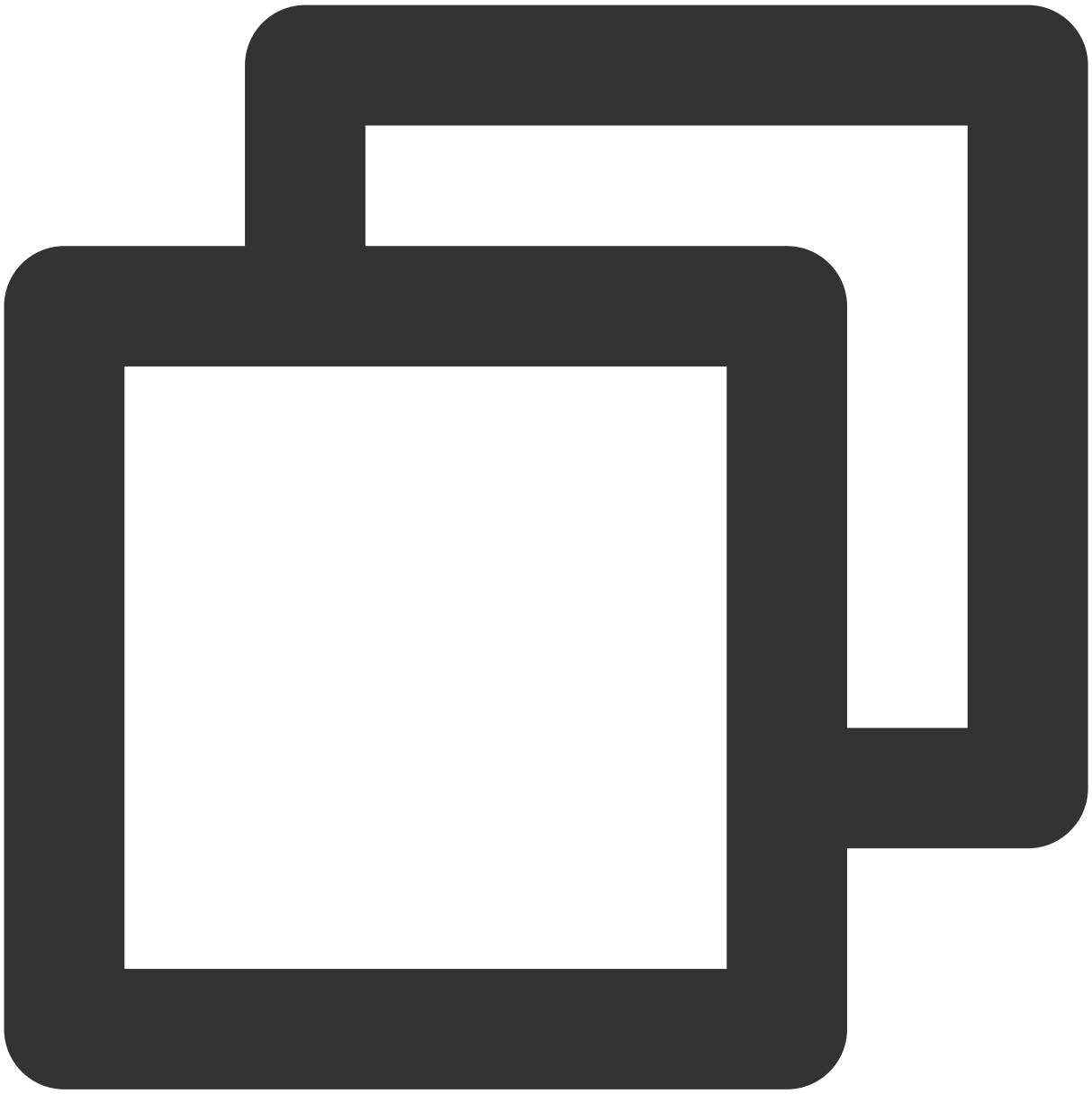


Set the message rewind parameters on the client.



```
endpoint='' // TDMQ for CMQ domain name
secretId ='' // User ID and key
secretKey = ''
account = Account(endpoint,secretId,secretKey)
queueName = 'QueueTest'
my_queue = account.get_queue(queueName)
queue_meta = QueueMeta()
queue_meta.rewindSeconds = 43200 // Message rewindable time in seconds
my_queue.create(queue_meta)
```

Using message rewind



```
my_queue.rewindQueue(1488718862) // Specify the time point for message rewind, whic
```

Dead Letter Queue

Last updated：2024-01-03 10:17:36

This document describes the basic concept, use cases, and usage of TDMQ for CMQ dead letter queue.

Feature Overview

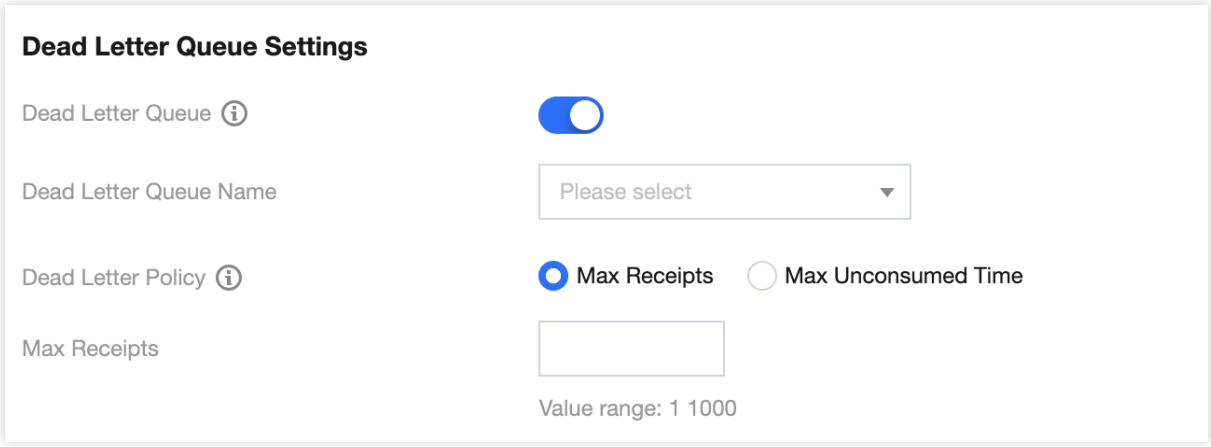
A dead letter queue (DLQ) is used to process messages that cannot be consumed normally. If consumption still fails after the maximum number of retries, it indicates that the consumer cannot consume a message under normal circumstances. At this point, TDMQ for CMQ will not immediately discard the message; instead, it will send the message to a special queue of the consumer, i.e., DLQ. You can enable DLQ for both new and existing queues.

Use Cases

Issue location: for example, if a message is not deleted after being consumed multiple times, this is generally because the message is not consumed properly, and there may be an issue which you should identify. You can set the maximum number of receipts after which the message will be placed in the specified DLQ for subsequent troubleshooting.
Priority queue: for example, O2O customers such as bike sharing operators have high requirements for access latency and real-timeness. In the bike unlocking logic, after TDMQ for CMQ retains 100 million messages, it will process the latest messages first and place old ones in the DLQ for consumption when the consumer is capable of consuming them.
As users may have been lost while old messages are waiting (such as bike unlocking through QR code scan), the value of old messages is low; therefore, we recommend processing the latest messages first.

Directions

When creating a queue in the TDMQ for CMQ console, you can enable the DLQ feature.



DLQ Queue Name: specify a queue as the DLQ of the queue.
Dead Letter Policy: select the method of triggering dead letter messages.
Maximum Number of Receipts: maximum number of times a message is allowed to be received before it is sent to the DLQ. The value range is 1–1000 times.
Maximum Unconsumed Time: maximum time a message remains unconsumed before it is sent to the DLQ. The value range is 5 minutes–12 hours.

Use Limits

A queue can be bound to only one DLQ.
The bound DLQ must be in the same region and under the same account as the queue.
If a DLQ has been bound to a queue, it cannot be deleted directly.
A queue with transaction message enabled cannot act as a DLQ.
A queue can be specified as the DLQ of up to 6 queues, but other queues cannot be specified as its DLQ (this helps avoid nesting).
A source queue (data queue produced to or consumed from by the client) can be unbound from the DLQ. The DLQ can then be deleted if it has no other queues bound to it.
If an existing regular queue is specified as a DLQ at time point T, and it contains unconsumed messages before time point T, all such messages need to be consumed before the dead letter policy can be triggered.

Topic Subscription
Topic Management

Last updated：2024-01-03 10:17:36

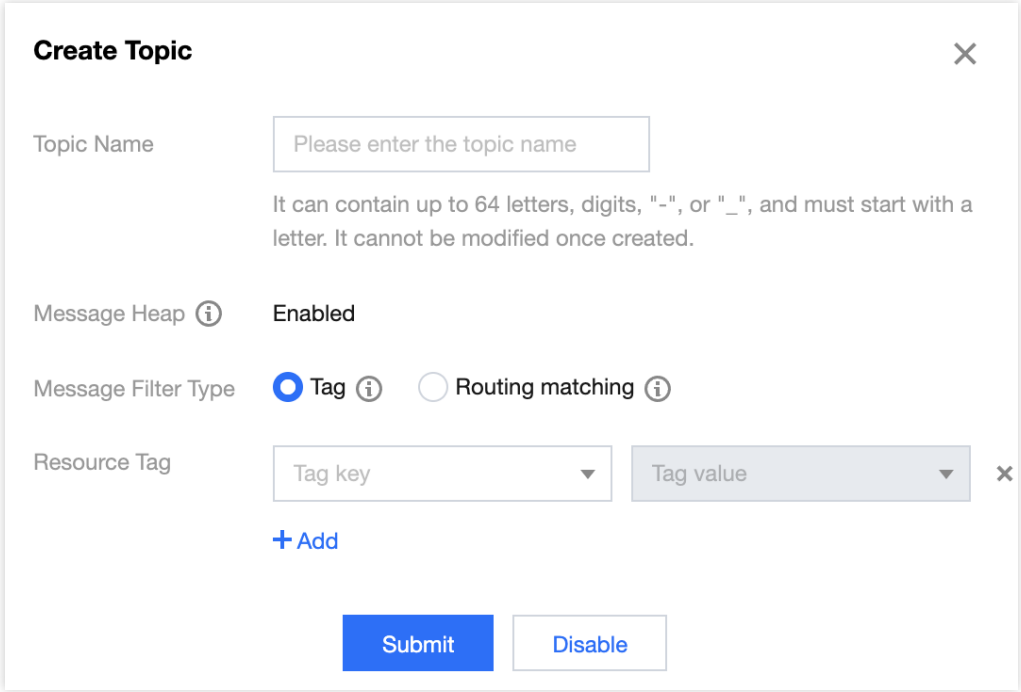
Overview

The topic model is similar to the publish/subscribe design pattern. A topic is the unit for sending messages, and subscribers under a topic are equivalent to observers. A topic will actively push published messages to subscribers. This document describes how to create or delete a topic in TDMQ for CMQ.

Directions

Creating topic

1. Log in to the TDMQ for CMQ console.
2. Select **Topic Subscription** on the left sidebar, select the region, click **Create**, and enter the information as prompted.



Topic Name: It can contain up to 64 letters, digits, "-", or "_", and must start with a letter. It cannot be modified once created.
Message Heap: Messages will be heaped temporarily if the message push is not triggered or the subscriber fails to receive them.
Message Filter Type:
Tag: TDMQ for CMQ can match message tags for production and subscription, which can be used for message filtering. For detailed rules, see Tag Key Matching Feature Description.
Routing Matching: The binding key and routing key are used together and are fully compatible with the topic match mode of RabbitMQ. The routing key carried when a message is sent is added by the client, and the binding key carried when a subscription is created is the binding relationship between the topic and the subscriber. For detailed rules, see Routing Key Matching Feature Description.
Resource Tag: It is optional and can help you easily categorize and manage TDMQ for CMQ resources in many dimensions. For detailed usage, see Managing Resource with Tag.

Deleting topic

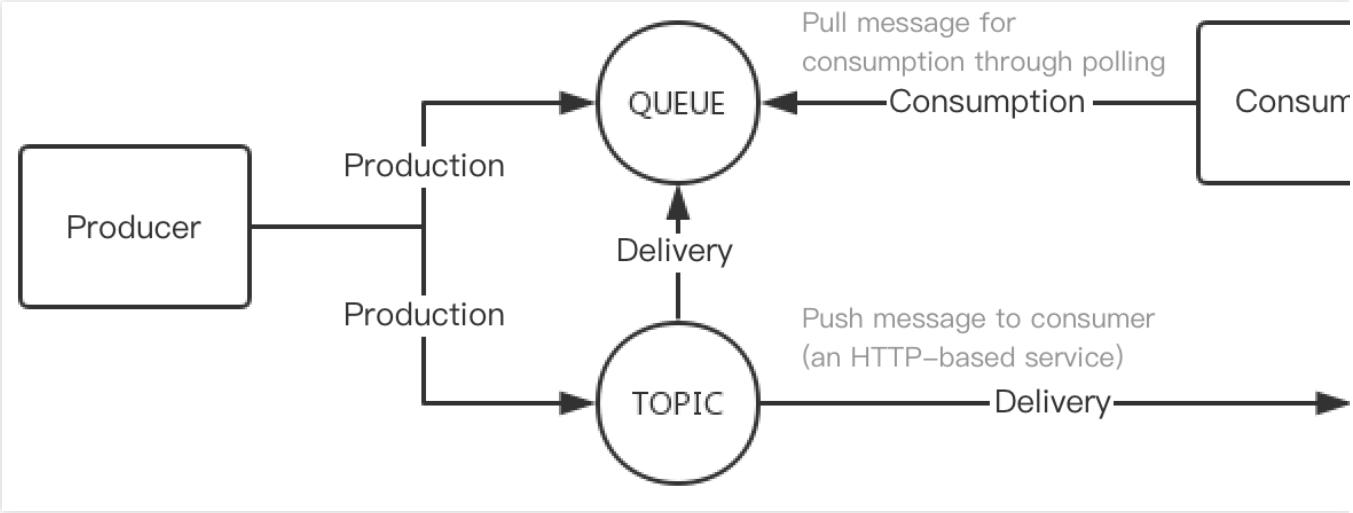
In the **topic** list, click **Delete** in the **Operation** column of the target topic to delete it. No messages will be pushed to this topic after it is deleted.

Subscription Management

Last updated：2024-01-03 10:17:36

Overview

A topic can publish messages only if it is subscribed to by at least one subscriber. If there are no subscribers, messages in the topic will not be delivered, and message publishing will be meaningless.
The model where a topic delivers a message to a subscriber is as shown below:



The topic follows the rules below when delivering the message to the subscriber:
The topic will try its best to deliver the message published by the producer to the subscriber.
If the delivery fails after multiple retries, the message will be retained in the topic and wait for the next delivery. If the next delivery still fails, the message will be discarded after the maximum lifecycle (1 day).
This document describes how to manage subscriptions under a topic in TDMQ for CMQ.

Prerequisites

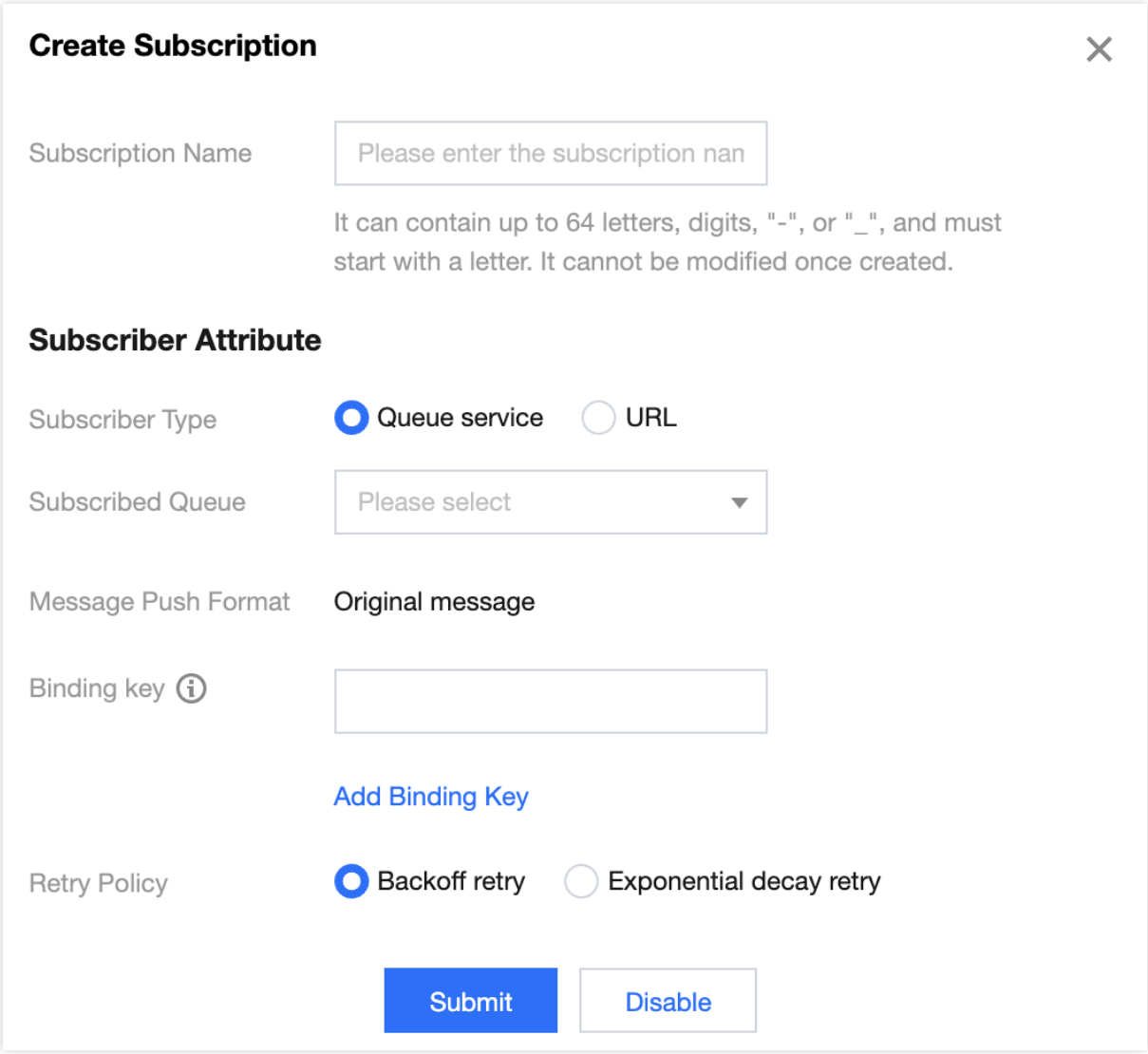
You have created a topic.

Directions

Creating subscriber

1. Log in to the TDMQ for CMQ console.

2. Select **Topic Subscription** on the left sidebar, select the region, and click the ID of the target topic to enter the topic details page.
3. Select the **Subscriber** tab at the top, click **Create**, and enter the subscriber information.



Subscriber Type
Queue Service: you can select a queue for the subscriber to use it to receive published messages.
URL: the subscriber can also process messages on its own without using a queue.
Subscriber Tag: when adding a subscriber, you can add filter tags (`FilterTag`), so that the subscriber can receive only messages with the specified tags. Up to 5 tags can be added for one subscriber. As long as a tag matches a topic filter tag, the subscriber can receive messages delivered by the topic. If a message does not have any tag, the subscriber cannot receive it.
Tag: for detailed rules, see Tag Matching Feature Description.
Routing Matching: for detailed rules, see Routing Key Matching Feature Description.

Retry Policy: after a message is published by a topic, it will automatically be pushed to the subscription. If the push fails, there are two retry policies:

Backoff retry: an attempt will be retried three times at random intervals between 10 and 20 seconds. After three retries, the message will be discarded for the subscriber and will not be retried again.

Exponential decay retry: an attempt will be retried 176 times at exponentially increasing intervals: 2^0 seconds, 2^1 seconds, ..., 512 seconds, 512 seconds, ..., 512 seconds. The total retry duration is 1 day. This is the default retry policy.

4. Click **Submit**, and you can see the created subscriber in the subscriber list.

Editing subscriber

On the **Subscriber** list page, click **Edit** in the **Operation** column to modify subscriber attributes.

Note:

You can only modify the subscriber's **message filter tag** and **retry policy**.

Deleting subscriber

On the **Subscriber** list page, click **Delete** in the **Operation** column to delete the subscriber.

Viewing Monitoring Data

Last updated：2024-01-03 10:17:36

Overview

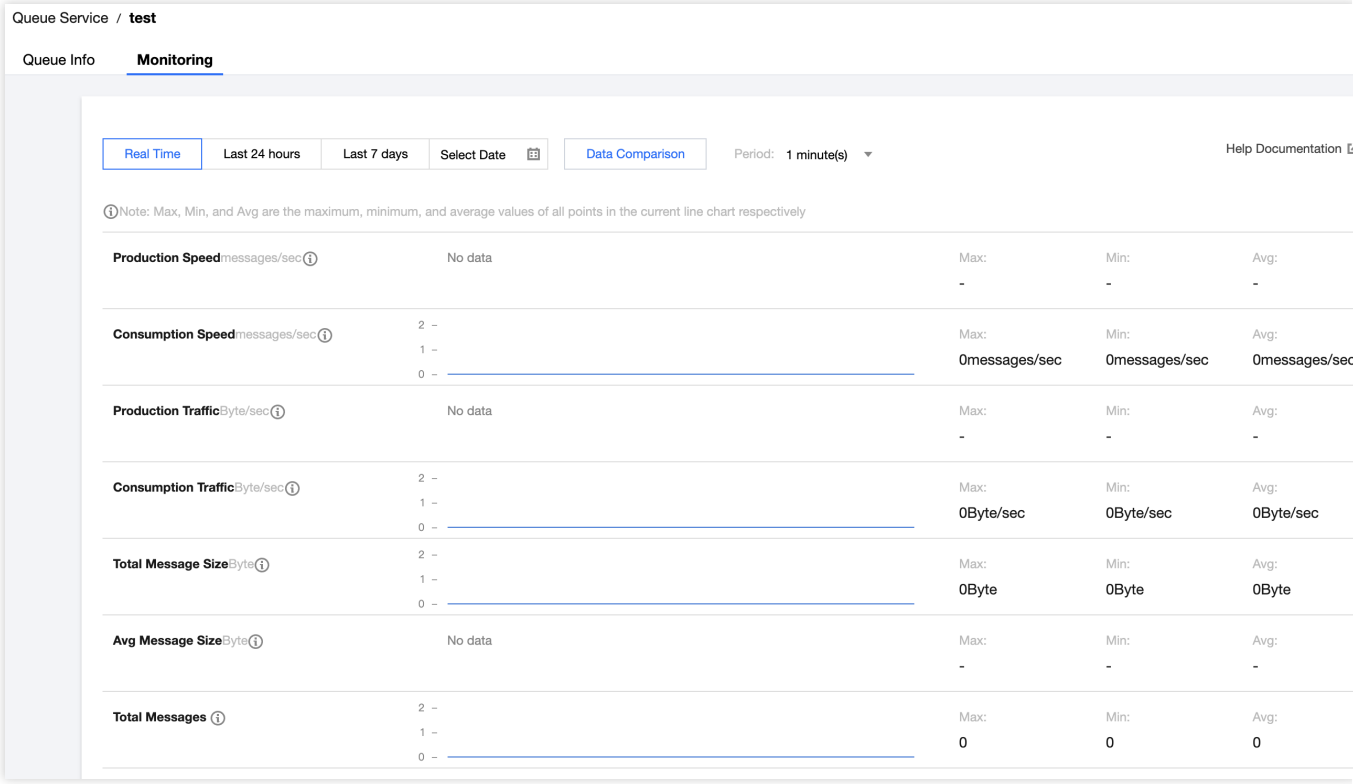
This document describes how to view the monitoring data of a created topic in the TDMQ for CMQ console.

Directions

1. Log in to the TDMQ for CMQ console.
2. Select **Topic Subscription** on the left sidebar, select the region, and click the ID of the target topic to enter the topic details page.
3. On the topic details page, select the **Monitoring** tab at the top.

Monitoring Metric Description

Monitoring Metric	Description
Production rate (messages/sec)	Number of messages produced by all producers under the topic per second in the selected time range
Consumption rate (messages/sec)	Number of messages consumed by all consumers under the topic per second in the selected time range.
Production traffic (bytes/sec)	Data size of messages produced by all producers under the topic per second in the selected time range
Consumption traffic (bytes/sec)	Data size of messages consumed by all consumers under the topic per second in the selected time range
Total message size (bytes)	The total data size of current messages
Average message size (bytes)	The average data size of current messages
Total number of messages	The total number of current messages
Number of producers	Number of producers connected to the queue
Number of subscribers	Number of consumers connected to the queue



Tag Key Matching Feature Description

Last updated：2024-01-03 10:17:36

The topic mode of TDMQ for CMQ supports subscription filtering by tag, which is similar to the "direct_routing" mode of RabbitMQ. The "topic_pattern" mode will be provided in the next iteration. As the usage policies of filter tags are complex, this document uses different scenarios as examples to describe them.

Scenario Description

Scenario 1

There are four subscribers A, B, C, and D and the message tags are as follows: `apple` for A, `xiaomi` for B, `imac+xiaomi` for C, and none for D.
A producer publishes 100 messages to the topic with message filter tags `apple` , `imac` , `iphone` , and `macbook` , and the topic is delivered to A, B, C, and D immediately.

Scenario analysis:

Subscriber	Message Tag	Message Receipt Description
A	apple	As `apple` can match the `apple` message filter tag, the 100 messages can be received properly.
B	xiaomi	No messages can be received.
C	imac+xiaomi	As `imac` can match the `imac` message filter tag, the 100 messages can be received properly.
D	-	All messages can be received.

Scenario 2

A topic has only four subscribers A, B, C, and D, none of which set any message tags.
A producer publishes 100 messages to the topic with message filter tags `apple` , `imac` , `iphone` , and `macbook` , and the topic is delivered to A, B, C, and D immediately.

Scenario analysis:

As none of subscribers A, B, C, and D have tags, messages do not need to be matched during delivery, and all of them can receive the 100 messages.

Scenario 3

A topic has only one subscriber A whose subscription tag is set to `xiaomi`.
A producer publishes 100 messages to the topic with message filter tags `apple`, `imac`, `iphone`, and `macbook`, and the topic is delivered to A immediately.

Scenario analysis:

As the subscription tag of subscriber A does not match, A cannot receive the 100 messages. In this case, the 100 messages will be discarded immediately and will not be retained in TDMQ for CMQ.
When the producer publishes to the topic, message filter tags can be set once only before publishing. They will be bound to message IDs and cannot be modified.

Scenario 4

A topic is named `test1`, and the subscription publishing API is called at 12:01 (this operation is named "publishing test1"). After the publishing, the topic is delivered to subscribers A, B, and C so as to deliver 200 messages to them. Suppose the result is as follows: A fails to receive 100 messages, B fails to receive 30 messages, and C successfully receives all the 200 messages.

Scenario analysis:

Message retention: among subscribers A, B, and C, suppose the total number of messages failed to be delivered is 110 (100 ones fail to be delivered to A, 30 to B, and none to C, and the failed ones may be repeated). As long as any subscriber is associated with a message, the message will not be unsubscribed from immediately.
Blocking policy: taking A as an example, the topic delivers 200 messages to A, and if the 101st message fails to be delivered, the delivery of subsequent 99 messages will be blocked. In this case, 100 messages will fail to be delivered.
Retry policy (backoff retry): the `test1` topic will deliver the messages again to A once every N seconds. Specifically, the failed 100 messages will be delivered to A again starting from the first one in sequence. If a message fails to be delivered for three consecutive times, it will be directly discarded, and the next message will be delivered, which may also be discarded after three failures, and so on.
Retry policy (decay exponential retry): taking A as an example, the topic delivers 100 messages concurrently to A (the sequence cannot be guaranteed). When a message fails to be delivered to A, the very message will be retried. If it fails again, subsequent messages will be blocked.
Inextensible message lifecycle: suppose the 110 messages are retained in the `test1` topic. No matter how many times they are retried for delivery, their lifecycle still remains 1 day. The time point when a message is pushed by the producer to the topic will be the lifecycle start time point, and the message will be deleted once the lifecycle expires.
Repush: the producer continuously produces new messages to the topic, and the subscription publishing API is called again at 12:02. Suppose the topic now has 210 messages (110 retained messages that fail to be delivered plus 100 messages that are produced in one minute) and delivers them again. In this case, as the retry policy for A and B is exponential decay retry, A and B cannot "respond", and the 210 messages will continue to be retained. C will receive only the 100 new messages.

Note:

Each message ID is used as the key, and the value is the associated subscribers, indicating whether consumption of each subscriber is successful.

Scenario 5

A topic named `test2` calls the subscription publishing API at 12:01, and this operation is named "publishing test2", which delivers 200 messages to subscribers A, B, and C.
Assume that A, B, and C successfully receive all the 200 messages.
Scenario analysis:
If the topic `test2` has only three subscribers A, B, and C, and the 200 messages are successfully consumed, it will immediately delete all these messages.

Rule Summary

You can summarize the following tag match rules from the aforementioned scenarios:

Subscriber	Whether Subscriber Has Tag	Whether Message Tag Exists	Message Receipt Description
A	Yes	No	The subscriber does not match and cannot receive messages.
B	No	Yes	The delivered messages do not need to be matched and all subscribers can receive messages.
C	Yes	Yes	Messages can be received only when the tags match. N:M match is supported; for example, if a message has 10 tags, and a subscriber has 4 tags, the subscriber can receive the message as long as there is one matching tag.
D	No	No	After a message is delivered, all subscribers can receive it.

Routing Key Matching Feature Description

Last updated：2024-01-03 10:17:36

TDMQ for CMQ routing key match is similar to exchange queue in RabbitMQ and can be used to filter messages so as to enable subscribers to get different messages by condition. When creating a topic, you can enable **Routing Matching Key**.

Use Instructions

The binding key and routing key are used together and function in a way similar to the message filtering capability of RabbitMQ. The routing key carried when a message is sent is added by the client, and the binding key carried when a subscription relationship is created is the binding relationship between the topic and the subscriber.

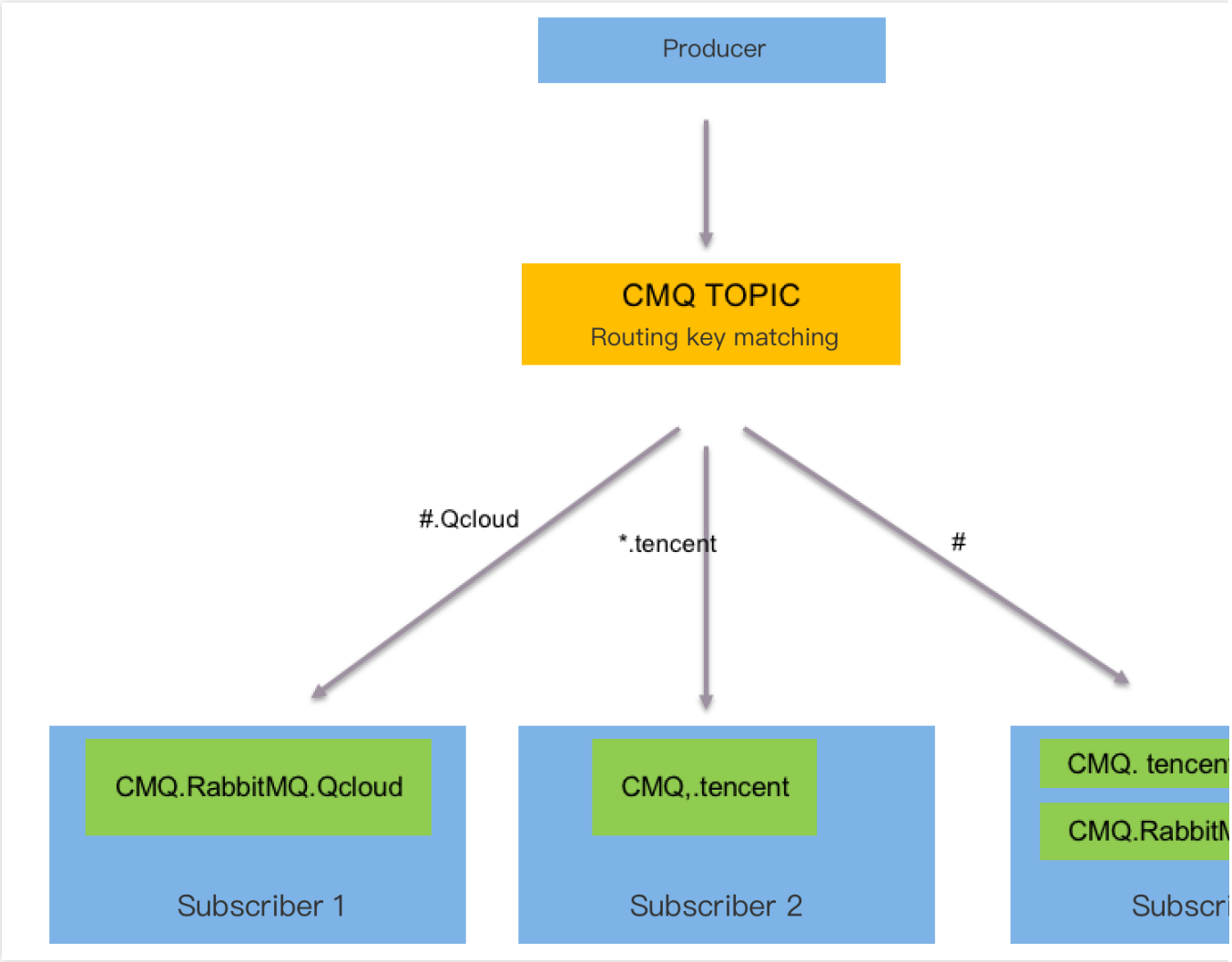
Use Limits

There can be up to 5 binding keys, and each of them can contain up to 64 bytes to indicate the route for message delivery, which can include up to 15 dots (namely up to 16 segments).
All routing keys are contained in a string, and each of them can contain up to 64 bytes to indicate the route for message delivery, which can include up to 15 dots (namely up to 16 segments).

Wildcard Description

`*` (asterisk) represents a word (a letter string) and cannot be empty.
`#` (hashtag) matches zero or multiple characters.

Example:
If the subscriber is **1.*.0**, and the message is **1.any character.0**, then it can be received by the subscriber.
If the subscriber is **1.#.0**, and the messages are **1.2.3.4.4.2.2.0** and **1.0**, then both of them can be received by the subscriber (the elements in the middle of the messages can be arbitrary).
If the subscriber is **#**, then the subscriber can receive all messages.



Access Management (CAM)

Last updated：2024-01-03 10:17:36

Basic CAM Concepts

The root account authorizes sub-accounts by associating policies. The policy setting can be specific to the level of **[API, Resource, User/User Group, Allow/Deny, and Condition]**.

Account

Root account: It owns all Tencent Cloud resources and can access any of its resources.
Sub-account: It includes sub-users and collaborators.
Sub-user: It is created and fully owned by a root account.
Collaborator: It has the identity of a root account. After it is added as a collaborator of the current root account, it becomes one of the sub-accounts of the current root account and can switch back to its root account identity.
Identity credential: It includes login credentials and access certificates. **Login credential** refers to a user's login name and password. **Access certificate** refers to Tencent Cloud API keys (`SecretId` and `SecretKey`).

Resource and permission

Resource: It is an object manipulated in Tencent Cloud services. TDMQ for CMQ resources include topics and queues.
Permission: It is an authorization that allows or forbids users to perform certain operations. By default, **a root account has full access to all resources under it**, while **a sub-account does not have access to any resources under its root account**.
Policy: It is a syntax rule that defines and describes one or more permissions. The **root account** performs authorization by **associating policies** with users/user groups.
View CAM documentation >>

Relevant Documents

Content	Document
Understand the relationship between policies and users	Concepts
Understand the basic structure of policies	Element Reference
Check CAM-enabled products	Overview

List of APIs Supporting Resource-Level Authorization

TDMQ for CMQ supports resource-level authorization. You can grant a specified sub-account the API permission of a specified resource.

APIs supporting resource-level authorization include:

API Name	API Description	Resource Type	Six-Segment Resource Example
ModifyCmqTopicAttribute	Modifies TDMQfor CMQ topic attributes	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
CreateCmqSubscribe	Creates a TDMQ for CMQ subscription	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
ModifyCmqSubscriptionAttribute	Modifies TDMQf for CMQ subscription attributes	Subscription	qcs::tdmq:${region}:uin/${uin}:subscr
RewindCmqQueue	Rewinds a TDMQf for CMQ queue	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
ModifyCmqQueueAttribute	Modifies TDMQ for CMQ queue attributes	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
ClearCmqSubscriptionFilterTags	Clears message subscription tags in TDMQ for CMQ	Subscription	qcs::tdmq:${region}:uin/${uin}:subscr
ClearCmqQueue	Clears messages in a TDMQ	Queue	qcs::tdmq:${region}:uin/${uin}:queue/

	for CMQ queue		
DeleteCmqSubscribe	Deletes a TDMQ for CMQ subscription	Subscription	qcs::tdmq:${region}:uin/${uin}:subscr
DeleteCmqTopic	Deletes a TDMQ for CMQ topic	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
BatchReceiveMessage	Consumes messages in batches	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
UnbindCmqDeadLetter	Unbinds a TDMQ for CMQ dead letter queue	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
DescribeCmqDeadLetterSourceQueues	Enumerates the source queues of a TDMQ for CMQ dead letter queue	Dead letter queue	qcs::tdmq:${region}:uin/${uin}:dlq/${s
DescribeCmqTopics	Enumerates all TDMQ for CMQ topics	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
DescribeCmqSubscriptionDetail	Queries TDMQ for CMQ subscription details	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
DescribeCmqQueues	Queries all TDMQ for CMQ queues	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
PublishCmqMsg	Sends a TDMQ for	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$

	CMQ topic message		
SendCmqMsg	Sends a TDMQ for CMQ message	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
DescribeCmqTopicDetail	Queries TDMQ for CMQ topic details	Topic	qcs::tdmq:${region}:uin/${uin}:topic/$
DescribeCmqQueueDetail	Queries TDMQ for CMQ queue details	Queue	qcs::tdmq:${region}:uin/${uin}:queue/
DeleteCmqQueue	Deletes a TDMQ for CMQ queue	Queue	qcs::tdmq:${region}:uin/${uin}:queue/

List of APIs Not Supporting Resource-Level Authorization

API Name	API Description	Six-Segment Resource
CreateCmqTopic	Creates a TDMQ for CMQ topic	*
CreateCmqQueue	Creates a TDMQ for CMQ queue	*

For APIs that do not support resource-level authorization, the `resource` field can be configured with an asterisk `*`.

Authorization Scheme Examples

Full access policy

Grant a sub-user full access to the TDMQ for CMQ queue service (for creating, managing, etc.).

1. Log in to the CAM console.

2. Click **Policy** on the left sidebar.
3. In the policy list, click **Create Custom Policy**.
4. In the **Select Policy Creation Method** pop-up window, select **Create by Policy Generator**.
5. On the **Edit Policy** page, click **Import Policy Syntax** in the top-right corner.
6. On the **Import Policy Syntax** page, search for **TDMQ**, select **QcloudTDMQFullAccess** in the search results, and click **OK**.
7. On the **Edit Policy** page, click **Next**, enter the policy name and description, and select the user/user group you want to associate.
8. Click **Complete**.

Read-only access policy

The following takes granting the read-only permission of a queue service as an example.
1. Log in to the CAM console.
2. Click **Policy** on the left sidebar.
3. In the policy list, click **Create Custom Policy**.
4. In the **Select Policy Creation Method** pop-up window, select **Create by Policy Generator** and enter the policy information.

Parameter	Description
Effect	Select **Allow**
Service	Select **TDMQ**
Action	Select **Read operation**
Resource	Select **Specific resources** and click **Add six-segment resource description** Region: Select the resource region Account: it is automatically populated Resource Prefix: queue Enter the name of the queue service you want to authorize
Condition	Allow access to specified operations only when the request is from the specified IP range

5. Click **Next**, enter the policy name and description, and select the user/user group you want to associate.
6. Click **Complete**.

Tag Management
Managing Resource with Tag

Last updated：2024-01-03 10:17:36

Overview

Tag is a key-value pair provided by Tencent Cloud to identify a resource in the cloud. It can help you easily categorize and manage TDMQ for CMQ resources in many dimensions such as business, purpose, and owner.

Note:

Tencent Cloud will not use the tags you set, and they are only used for your management of TDMQ for CMQ resources.

Use Limits

You need to pay attention to the following use limits of tags:

Limit	Description
Quantity	One Tencent Cloud resource can have up to 50 tags.
Tag key	You cannot place `qcloud` , `tencent` , or `project` at the beginning of a tag key as they are reserved by the system. A tag key can contain up to 255 digits, letters, and special symbols (`+=.@-`).
Tag value	It can contain up to 127 digits, letters, and special symbols (`+=.@-`) or be an empty string.

Directions and Use Cases

Use case

A company has 6 TDMQ for CMQ queues, with the department, business scope, and owner information as described below:

Queue ID	Department	Business Scope	Owner
cmqq-372ovdpw8ob1	Ecommerce	Marketing	John
cmqq-372ovdpw8ob2	Ecommerce	Marketing	Harry

cmqq-372ovdpw8ob3	Gaming	Game A	Jane
cmqq-372ovdpw8ob4	Gaming	Game B	Harry
cmqq-372ovdpw8ob5	Entertainment	Post-production	Harry
cmqq-372ovdpw8ob6	Entertainment	Post-production	John

You can add the following three tags to the `cmqq-372ovdpw8ob1` queue:

Tag Key	Tag Value
dept	ecommerce
business	mkt
owner	zhangsan

Similarly, you can also set appropriate tags for other resources based on their department, business scope, and owner information.

Setting tag in TDMQ for CMQ console

After designing the tag keys and values as detailed above, you can log in to the TDMQ for CMQ console to set tags.

1. Log in to the TDMQ for CMQ console.

2. On the **Queue Service** page, select the target region and queue and click **Edit Resource Tag** at the top of the page.

3. Set tags in the **Edit Tag** pop-up window.

For example, add three tags for the `cmqq-372ovdpw8ob1` queue.

Note:

If existing tags cannot meet your needs, go to Tag Management to create more.

4. Click **OK**, and you will be prompted that the tags have been modified successfully. You can view the tags bound to a queue in its **Resource Tag** column.

Filtering resource by tag key

You can filter out queues bound to a specific tag in the following steps:

1. Select **Tag** in the search box at the top-right corner of the Queue Service page.

2. In the window that pops up, select the tag you want to search for and click **OK**.

For example, if you select `Tag: owner:zhangsan`, you can filter out queues bound to the tag key `owner:zhangsan`.

Editing Tag

Last updated：2024-01-03 10:17:36

Overview

This document describes how to edit resource tags.

Use Limits

For the use limits of tags, see Managing Resource with Tag - Use Limits.

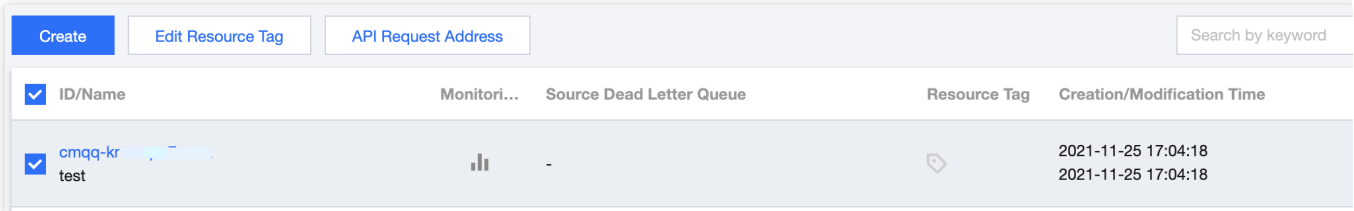
Prerequisites

You have logged in to the TDMQ for CMQ console.

Directions

Take modifying a tag bound to a queue as an example:

1. On the **Queue Service** list page, select the target region and queue and click **Edit Resource Tag** at the top of the page.



Note:

You can batch edit tags for up to 20 resources at a time.

2. In the **Edit Tag** pop-up window, add, modify, or delete tags as needed.

Use Cases

For directions on how to use tags, see Managing Resource with Tag.

Alarm Configuration

Last updated：2024-01-03 10:17:36

Overview

Tencent Cloud provides the Cloud Monitor service for all users by default; therefore, you do not need to manually activate it. Cloud Monitor will start collecting monitoring data only after a Tencent Cloud product is used.
TDMQ for CMQ allows you to monitor the resources (topics and queues) created under your account, so that you can keep track of the status of your resources in real time. You can configure alarm rules for monitoring metrics. When a monitoring metric reaches the set alarm threshold, Cloud Monitor will notify you of exceptions in time via the notification channels you specified.

Directions

Configuring alarm policy

An alarm policy can determine whether an alarm notification should be sent based on the comparison between the monitoring metric and the given threshold in the selected time period. You can promptly take appropriate precautionary or remedial measures when the alarm is triggered by a TDMQ for CMQ status change. Properly configured alarm policies help improve the robustness and reliability of your applications.

Note:

Be sure to configure alarms for your instance to prevent exceptions caused by traffic spikes or specification limits.

1. Log in to the CM console.
2. On the left sidebar, select **Alarm Configuration** > **Alarm Policy** and click **Create**.
3. On the **Alarm Policy** page, select a policy type and instance and set the alarm rule and notification template.

Monitoring Type: select **Cloud Product Monitoring**.

Policy Type: select **TDMQ alarm** > **CMQ**.

Alarm Object: select the TDMQ for CMQ resource for which to configure the alarm policy.

Trigger Condition: you can select **Select template** or **Configure manually**. The latter is selected by default. For more information on manual configuration, see the description below. For more information on how to create a template, see Creating trigger condition template.

Note：

Metric: for example, if you select 1 minute as the statistical period for the "message retention volume" metric, then if the message retention volume exceeds the threshold for N consecutive data points, an alarm will be triggered.

Alarm Frequency: for example, "Alarm once every 30 minutes" means that there will be only one alarm triggered every 30 minutes if a metric exceeds the threshold in several consecutive statistical periods. Another alarm will be triggered

only if the metric exceeds the threshold again in the next 30 minutes.

Notification Template: you can select an existing notification template or create one to set the alarm recipient objects and receiving channels.

4. Click **Complete**.

For more information on alarms, see Creating Alarm Policy.

Creating trigger condition template

1. Log in to the CM console.
2. On the left sidebar, click **Trigger Condition Template** to enter the **Template** list page.
3. Click **Create** on the **Trigger Condition Template** page.
4. On the **Create Template** page, configure the policy type.

Policy Type: select **TDMQ alarm** > **CMQ**.

Use preset trigger condition: select this option and the system recommended alarm policy will be displayed.

5. After confirming that everything is correct, click **Save**.
6. Return to the **Create Alarm Policy** page, click **Refresh**, and the alarm policy template just configured will be displayed.

Message Query and Trace

Last updated：2024-07-10 16:13:18

When a message is sent from the producer to the TDMQ for CMQ server, and then the consumer pulls and consumes it from the server, TDMQ for CMQ will record the related information and the process flow of the message, and present it as a message record and message trace in the console.

Overview

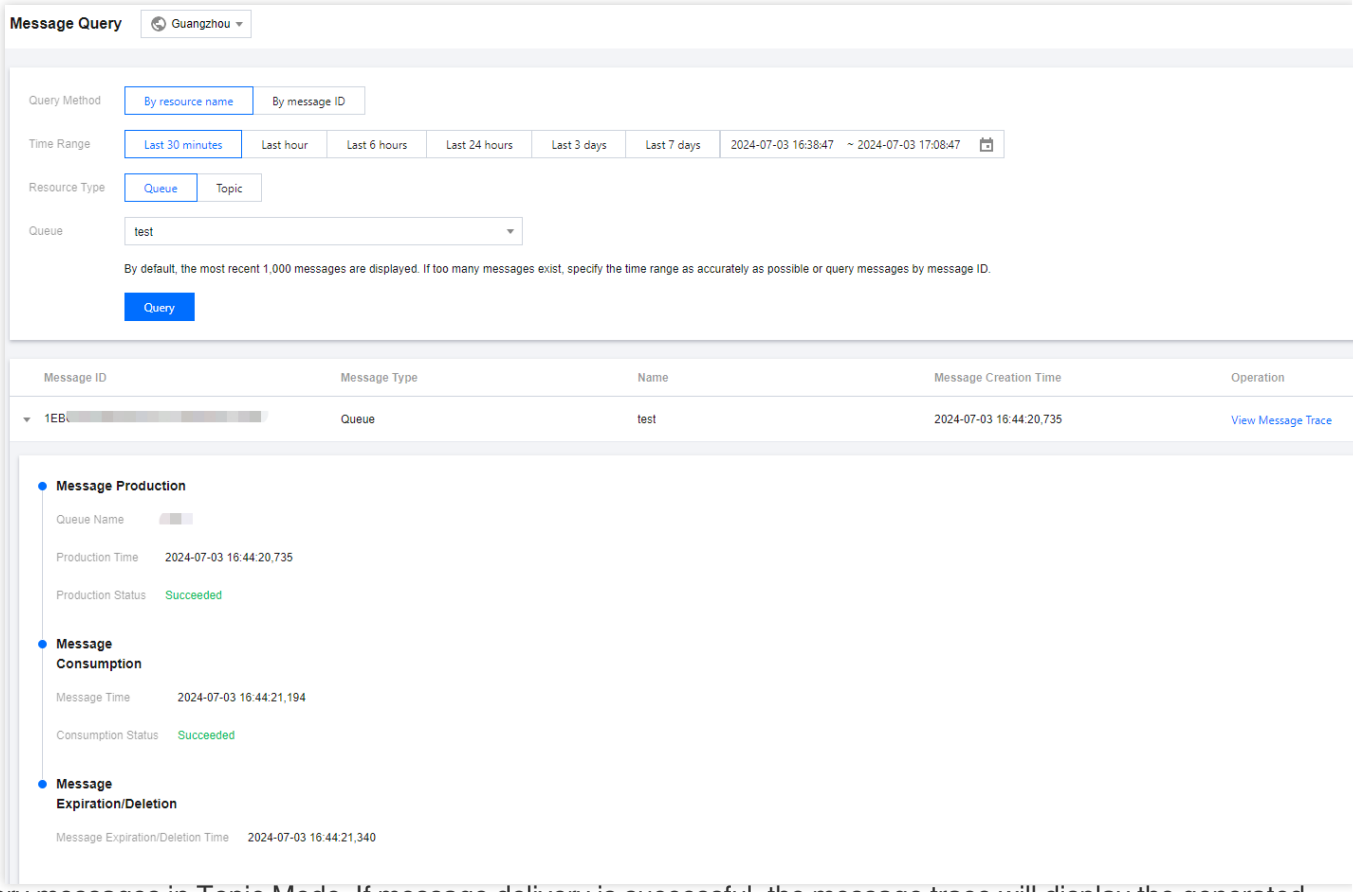
When you need to troubleshoot the following issues, you can use the message query feature in the TDMQ for CMQ console to view the details and message traces of a specific message by Resource & Time Dimension or directly by message ID.
View whether the message was sent successfully and the exact time it arrived at the server.
View the lifecycle of the message and whether it was deleted by the server.
View the consumers who consumed the message, whether it was consumed successfully, and the exact time it confirmed consumption.
It is required to analyze the performance of the distributed system. View the processing latency of related messages by MQ.

Query Limits

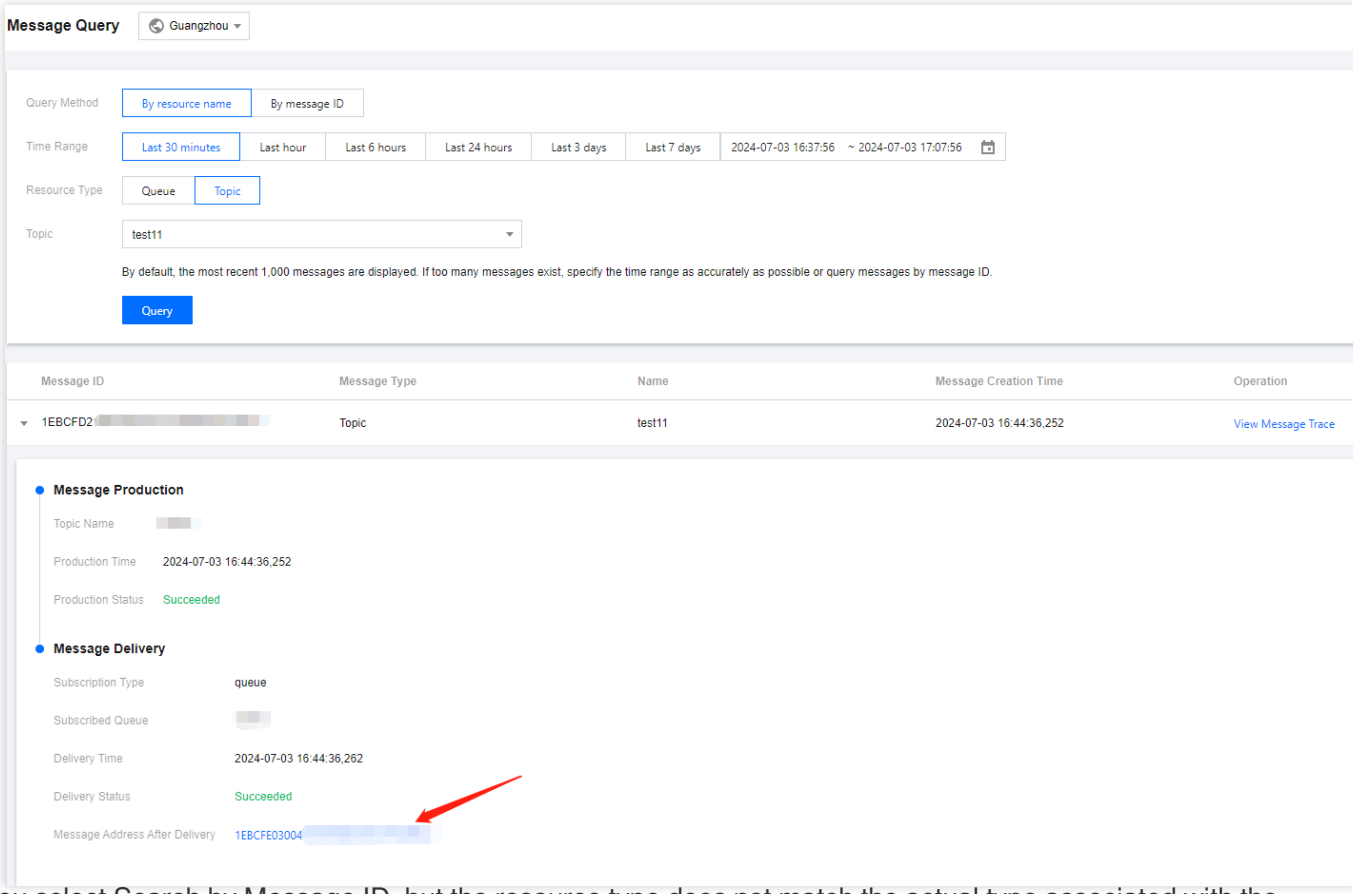
Message query can retrieve messages from the past 7 days at most.
You can query up to 1,000 messages at a time.

Directions

1. Log in to [TDMQ for CMQ Console] (https://console.tencentcloud.com/tdmq/cmq-queue?rid=1!3420c0e1942b569422ee07a5145ce9c7), and click **Message Query** on the left sidebar.
2. On the Message Query page, first select the region and environment, and then select the way based on the actual situation (click **By Resource Name** and select **Time Range** or click **By Message ID**) to query. If you know the specific Message ID, it is recommended that you enter it for a precise query.
3. Select **Resource Type** and **Queue** as needed, click **Query**, and the list below will display all the queried results and paginate them, as shown in the following figure.
4. After clicking **View Message Trace**, you can view the lifecycle of the current message (such as Production Time, Consumption Time, and Expiration Time).



5. Query messages in Topic Mode. If message delivery is successful, the message trace will display the generated message ID, which can be directly clicked to navigate to the trace query of the new message.



6. If you select Search by Message ID, but the resource type does not match the actual type associated with the message, you will be prompted to select the correct resource type as shown in the following figure.