Alibaba Cloud Table Store

Data Models

Issue: 20190613

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use**

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*

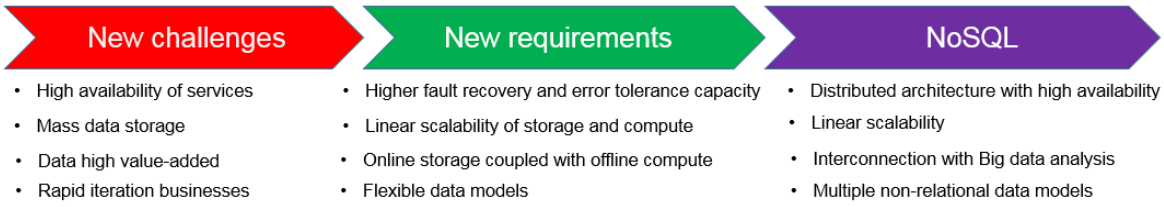
Style	Description	Example
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich {stand | slave}`

Contents

1 Preface

Table Store is an Alibaba Cloud proprietary distributed NoSQL database. Nowadays , many application systems do not solely rely on relational databases. Instead, they select different database types depending on the business scenarios they are deployed in. For example, cached KeyValue data is stored in Redis, file type data is stored in MongoDB, and graph data is stored in Neo4J.

Traditional relational databases have difficulties in carrying data at such large scales , so a distributed database with high scalability is required. However, it is difficult to develop a highly available and highly scalable distributed database based on the traditional relational data model. If it is possible to use a model that is simpler than the relational data model and has simplified constraints and increased availability and scalability, business needs can be effectively met. It is based on these ideas that NoSQL was designed and developed.



NoSQL has the following characteristics:

- Multi-data models

  Many types of data models, such as KeyValue, Document, Wide Column, Graph , and Time Series, are provided to fulfill different data requirements. NoSQL databases operate far beyond the constraints of the relational data model and create a diversified business direction. These multi-data models are more suitable for all kinds of real-world scenarios.

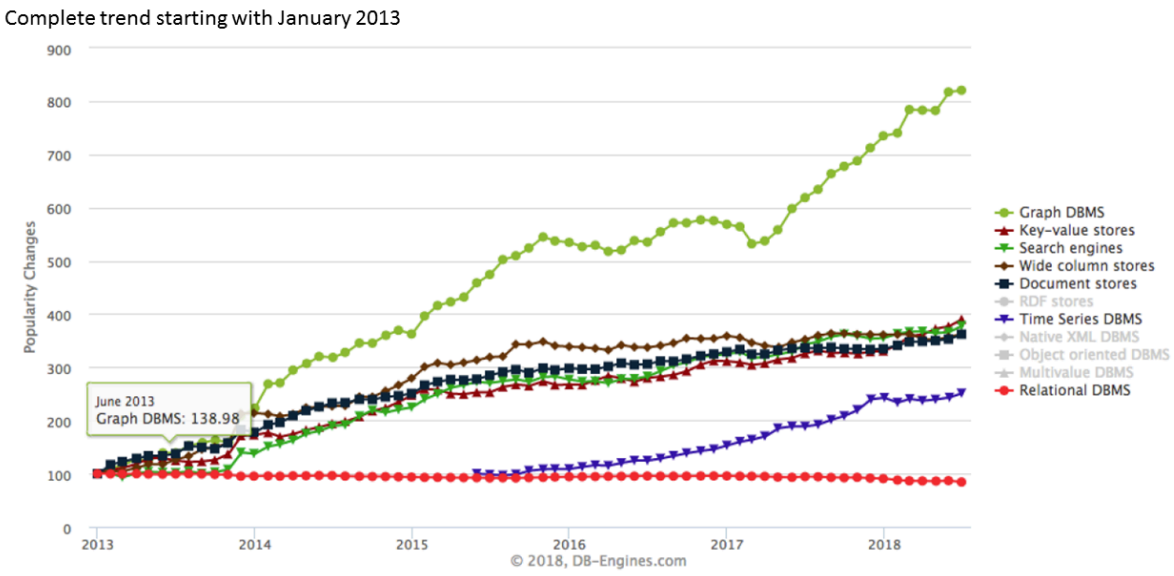
- High concurrency and low latency

  NoSQL is designed to provide high-concurrency and low-latency access to online businesses.

- High scalability

  Scalability is one of the core design goals of NoSQL to cope with the explosive growth in volume of data.

DB-Engines aims at collecting information from DBMS (Database Management System) and development trend of multiple NoSQL databases from 2013 to 2018 is as follows:



It can be seen that all types of NoSQL databases have thrived in recent years. Alibaba Cloud Table Store, as a distributed NoSQL database, adopts a multi-model architecture and supports both Wide Column and Timeline architectures.

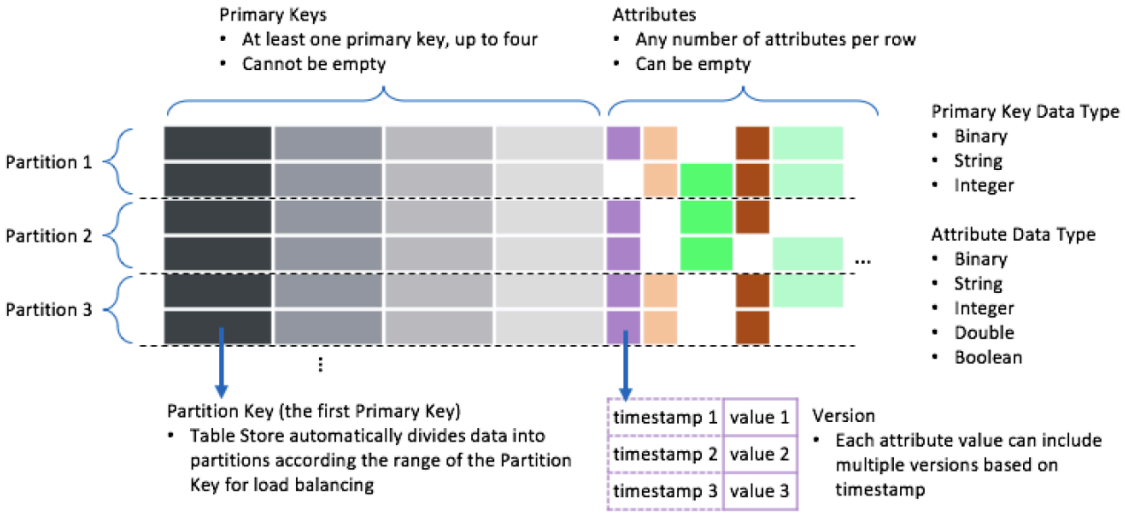
The Wide Column model, first put forward by Bigtable, has become the classic model among widely-used systems of the same type. Currently most semi-structured and structured data is stored in this model system. In addition to the Wide Column model, we have introduced a completely new data model called Timeline. The Timeline model is a new generation of model for messaging data. It is already widely used in instant messaging (IM), feeds, Internet of Things (IoT) equipment, and other systems where messaging data has to be stored and synchronized. These two models are described as follows:

2 Wide Column

2.1 Introduction

The Wide Column model differs from the relational model in the following aspects:

- The characteristics of Wide Column are: three-dimensional (rows, columns, and time), schema-free, wide columns, multi-version data, and TTL management.
- The characteristics of the relational model are : two-dimensional (rows and columns) and fixed schema.



The Wide Column model consists of the following parts:

- Primary key: Every row has a primary key with a multi-column structure (1-4 columns). The primary key is defined as a fixed schema, and is used primarily to uniquely distinguish a row of data.
- Partition key: The first column of the primary key is called a partition key. The partition key is used to partition the table by range. Every partition is distributively dispatched to services on different machines. Within the same partition key, we provide cross-row transactions. For more information, see *Primary key and attribute*.
- Attribute column: In one row, with the exception of the primary key, all other columns are attribute columns. Attribute columns correspond to many values. Different values correspond to different versions, and each row stores an unlimited number of attribute columns.

- Version: Each value corresponds to a different version that acts as a timestamp to define the time to live of that data.
- Data type: Table Store allows many different data types, including String, Binary, Double, Integer and Boolean.
- Time To Live (TTL): Each table defines the amount of time a data can be stored before being deleted. For example, if the TTL is defined as one month, the data written into the table more than a month ago will be cleared automatically. The write time of the data is determined by the version number. This write time is usually taken from the server time, but it can also be determined by the time specified by the application. For more information, see *Data versions and Time To Live*.
- Max versions: Each table defines the maximum number of version data that can be stored in a column, which is used to control the number of versions in each column. If the number of versions in an attribute column exceeds the value in max versions, the earliest version is deleted.

2.2 Primary keys and attributes

In Table Store, tables, rows, primary keys, and attributes are the core components that you work with. A table is a collection of rows, and each row consists of a primary key and attributes. The first column of a primary key is called the partition key.

Primary keys

Primary keys are used to uniquely identify each row in a table. A primary key is a combination of one to four attributes. When creating a table, you must specify the composition of the primary key, including the name of each attribute, the data type of each attribute, and the sorted order of attributes. In Table Store, you can specify a data type, such as String, Binary, or Integer, for an attribute.

Table Store indexes data of a table based on the primary key of the table. All rows of the table are sorted in ascending or descending order based on the primary key.

Partition keys

The first column of a primary key is called the partition key. Table store assigns a row of data to the corresponding partitions determined by the range of each row's partition keys to achieve load balancing. Rows that have the same partition key value belong to the same partition. A partition may store rows with multiple partition

key values. Table Store separates a partition or merges multiple partitions based on specific rules. This process is completed automatically.

Note:
The partition key is used as the minimum partition unit. Data under the same partition key value cannot split further. To prevent partitions from being too large to split, we recommend that the total size of all rows with the same partition key value is less than 10 GB.

Attributes

A row consists of multiple attributes. The number of attributes for each row is not restricted, which means that each row has a different number of attributes. The value of an attribute of a row can be null. The values of an attribute in multiple rows can be of different data types.

An attribute includes the version property. Multiple versions of attribute values can be retained as required for querying or other uses. Additionally, data in an attribute has its own TTL. For more information, see *Data versions and life cycle*.

2.3 Read/write throughput

The read/write throughput is measured by read/write capacity units (CUs), which is the smallest billing unit for the data read and write operations.

- One read CU indicates that 4 KB data is read from the table.
- One write CU indicates that 4 KB data is written into the table.
- Data smaller than 4 KB during the operation is rounded up to the nearest CU. For example, writing 7.6 KB data consumes two write CUs, and reading 0.1 KB data consumes one read CU.

When applications use an API to perform Table Store read/write operations, the corresponding amount of read/write CUs is consumed.

Reserved throughput

The reserved read/write throughput is an attribute of a table. When creating a table, the application specifies the read/write throughput reserved for the table. Configuring the reserved read/write throughput does not affect the table's access performance and service capability.

For reserved throughput billing, the reserved throughput value is always used to calculate the hourly fee even if an application consumes less than the specified amount of throughput.

For example, suppose that an application reads 3 KB of data per record and 80 records per second from a table. In this case, the application consumes 80 capacity units per second.

If you set the reserved read throughput to 80 capacity units per second, the hourly fee is calculated by using the following formula: Hourly Fee = 80 reserved read throughput capacity units x Hourly Price for Reserved Read Throughput. It is enough for 288000 (80 x 3600 seconds) reads per hour.

 Note:

- Reserved read/write throughput can be set to zero.
- When the reserved read/write throughput is greater than zero, Table Store assigns and reserves enough resources for the table according to this configuration to guarantee low resource costs.
- For a non-zero reserved read/write throughput, your Table Store service is billed even if no read and write requests are made. To guarantee billing accuracy, Table Store limits the maximum reserved read/write throughput to 5000 CUs per table (neither read throughput nor write throughput can exceed 5000 CUs). If you require more than 5000 CUs of reserved read/write throughput for a single table, *Open a ticket* to increase the throughput.
- The reserved read/write throughput of a non-existent table is regarded as zero. To access a non-existent table, one additional read CU or one additional write CU is consumed depending on the actual operation.

Applications dynamically modify the reserved read/write throughput configuration of the table through the UpdateTable operation.

Additional throughput

The additional read/write throughput refers to the portion of the actual consumed read/write throughput that exceeds the reserved read/write throughput. Its refresh interval is one second.

In the following example, the reserved read throughput is set to 100 units. T0, T1, and T2 show the reserved read throughput and the additional read throughput that an application consumed in three consecutive seconds:

- T0: The actual read throughput consumption is 120 units. The consumption of the reserved read throughput and the consumption of the additional read throughput are 100 units and 20 units, respectively.
- T1: The actual read throughput consumption is 95 units. The consumption of the reserved read throughput and the consumption of the additional read throughput are 100 units and 0 units, respectively.
- T2: The actual read throughput consumption is 110 units. The consumption of the reserved read throughput and the consumption of the additional read throughput are 100 units and 10 units, respectively.

In the three consecutive seconds, the consumption of the reserved read throughput is 100 units, and the total consumption of the additional read throughput is 30 units.

 Note:
Table Store uses the average value per hour to calculate the consumption of the reserved throughput and uses the total amount per hour to calculate the consumption of the additional throughput.

For the additional read/write throughput mode, it is difficult to estimate the amount of compute resources that need to be reserved for data tables. Table Store is required to provide sufficient service capability to effectively handle access traffic spikes. For this reason, the unit price of additional read/write throughput is higher than that of reserved read/write throughput. To make sure that low costs are maintained, we recommend that you set an appropriate value of the reserved read/write throughput.

 Note:
Because it is difficult to accurately reserve resources based on the additional read/write throughput, in extreme situations, Table Store may return an error OTSCapacityUnitExhausted to an application when an access to a single partition key consumes 10,000 CUs per second. In this case, policies such as backoff retry are used to reduce the frequency of access to the table.

For more information, see *Table Store tables* and *billing methods*.

3 Timeline

Overview

The Timeline model is a data model designed for message data scenarios. The model supports some special requirements of message data scenarios, such as message order preservation, storage of large numbers of messages, and real-time synchroniz ation. The model also supports the full-text search and bool query. The model is applicable to message scenarios such as instant messaging (IM) and Feed streams.

Architecture

The Timeline model provides clear core modules in a simple design. You can easily use this model, and set the model according to your business. The architecture of the model includes the following components:

- Store: a store of Timeline data. The store is similar to a table in a database.
- Identifier: an identifier used to identify Timeline data.
- Meta: the metadata used to describe Timeline data. The metadata is stored in a free -schema structure and can contain any column.
- Queue: stores all messages in a Timeline.
- SequenceId: the serial number of a message body in the Queue. The SequenceId values must be incremental and unique. The Timeline model generates SequenceId values by using an auto-increment column. You can also specify SequenceId values by manual.
- Message: the message body in the Timeline. The message is stored in a free-schema structure and can contain any column.
- Index: includes Meta Index and Message Index. You can customize indexes for any columns in Meta or Message to provide the bool query.

Features

The Timeline model supports the following features:

- Manages Meta data and messages, including basic data operations such as create, read, update, and delete.
- Supports the bool query and full-text search for Meta data and messages.

- Generates SequenceId values in two ways: auto-increment column and manual setting.
- Supports the Timeline Identifier that contains multiple columns.
- Compatible with the Timeline 1. X model. The TimelineMessageForV1 example of the Timeline model can directly read messages from and write messages to the V1 version.

Timeline

```
< dependency >
    < groupId > com . aliyun . openservic  es . tablestore </ groupId
 >
    < artifactId > Timeline </ artifactId >
    < version > 2 . 0 . 0 </ version >
</ dependency >
```

Table Store Java SDK (integrated with the Timeline model)

```
< dependency >
  < groupId > com . aliyun . openservic  es </ groupId >
  < artifactId > tablestore </ artifactId >
  < version > 4 . 12 . 1 </ version >
</ dependency >
```