

# 数蚕时序数据库

(集群版) 指导教程

公 司: 北京数蚕科技有限公司  
版 本: 0.2.2  
作 者: 肖堂  
时 间: 2024-02-21



## 说 明

这是一份由北京数蚕科技有限公司 (以下简称**数蚕**) 官方制作的针对数蚕时序数据库操作的指导教程。本教程前部分是为数蚕时序数据库的安装和日常使用; 后部分内容是数据库的深度原理设计方法和结构分析。希望可以为时序数据库使用及研发人员提供便利的快速实践方法。

本教程由**时序数据库安装指南**、**配置说明**、**数据操作**、**命令规范**、**数据库原理**、**开发及维护**七个部分组成。本教程适合数据库管理员、数据库研发人员及数据分析人员使用。

注意: 本教程是针对时序数据库 (集群版)v0.2.2 版本的技术手册, 其它版本的产品请以时序数据库产品附加手册或**官方网站**对应版本产品文档为参考。

# 目录

<b>1 安装指南</b>	<b>5</b>
1.1 基础环境准备	5
1.2 集群服务	5
1.2.1 安装集群服务	5
1.2.2 运行集群服务	6
1.2.3 关闭集群服务	6
1.2.4 卸载集群服务	6
1.3 时序数据库	6
1.3.1 安装时序数据库服务	6
1.3.2 运行时序数据库服务	7
1.3.3 关闭时序数据库服务	8
1.3.4 卸载时序数据库服务	8
<b>2 配置说明</b>	<b>9</b>
2.1 集群配置文件	9
2.2 时序数据库配置文件	10
<b>3 数据操作</b>	<b>12</b>
3.1 对象和概念	12
3.2 数据表	12
3.3 库操作	12
3.4 限制说明	12
<b>4 命令规范</b>	<b>14</b>
4.1 数据交换格式规范	15
4.2 命令规范	15
4.2.1 库命令	15
4.2.2 表命令	16
4.2.3 数据写操作	17
4.2.4 数据更新操作	18
4.2.5 数据读操作	18
4.2.6 时间点规范	18
<b>5 数据库原理</b>	<b>20</b>
5.1 命令执行原理	20
5.2 数据存储原理	20
5.3 存储数据块	20
5.4 集群连接原理	20
5.5 分布式存储原理	21
<b>6 开发</b>	<b>22</b>
6.1 开发包安装和部署	22
6.2 基础开发流程	22
6.3 语言相关	23
6.3.1 c	23
6.3.2 c++	24
6.3.3 java	24
6.3.4 python	24
6.3.5 其它语言或版本	24
6.4 常见问题	24

<b>7 维护</b>	<b>25</b>
7.1 集群管理	25
7.1.1 集群备份和扩展	25
7.1.2 集群异常处理	25
7.2 时序数据库管理	25
7.3 时序数据库备份和恢复	25
7.4 时序数据库异常处理	25
7.5 产品更新和升级	26
7.6 商业技术支持或服务	26

# 1 安装指南

安装数蚕时序数据库需要先准备基础的软硬件环境。然后安装对应的程序压缩包，并运行服务或客户端。同时可以针对不同业务场景进行个性化的配置。

## 1.1 基础环境准备

数蚕时序数据库服务端运行环境要求:

CPU 要求:

可运行于 x86/amd64, aarch64 及 arm 指令的其它 cpu 架构。特殊的场景的 cpu 框架可能需要联系官方提供对应架构版本。

内存空间要求: 根据业务查询数据量内存存储空间要求大小不同, 建议至少 4G 有效使用内存。

硬盘空间要求:

系统安装最小空间: 1G 根据业务数据量大小存储空间大小不同, 建议至少 60G 的空闲存储空间。

操作系统: 数蚕时序数据库支持多数系统版本, 可以针对的需要下载不同的系统版本, 当前官方可以提供以下系统环境的发行版。

```
almalinux 9
debian 11
mint 21
fedora 36
manjaro 21
mac 12
oracle linux 9
redhat enterprise linux 9
suse linux enterprise 15
ubuntu 22
windows 7
windows 10
windows 11
windows server 2016
windows server 2019
windows server 2022
freebsd 13
```

数蚕时序数据库 GUI 客户端支持 windows 7/10/11, freebsd 13, debian 11。

## 1.2 集群服务

### 1.2.1 安装集群服务

数蚕时序数据库集群版需要运行数据存储集群服务。集群服务运行包正常位于程序安装包 cluster 目录中。

完整的集群系统目录和文件作用如下:

文件名	说明
tdb	集群时序数据文件存储目录
sct_dt_cluster_s	数蚕集群服务端执行程序
node.toml	数蚕集群节点配置文件

安装集群通过以下两个步骤:

- 1, 准备存储的服务器节点机器。并根据业务需要规划分配各机器存储使用空间。
- 2, 拷贝 cluster 目录中集群服务程序到各目标服务器中。并在配置文件中分配各节点数据空间 (详见配置说明中集群配置文件说明)。

### 1.2.2 运行集群服务

运行集群服务。在 windows 平台运行 sct\_dt\_cluster\_s.exe 或使用目录中安装脚本 Install\_Service.bat 安装为系统服务。linux 系列平台运行 sct\_dt\_cluster\_s 或安装为系统服务。

集群正常运行时主节点应该出现节点空间分配信息，各节点显示当前分配空间信息和运行端口。

### 1.2.3 关闭集群服务

退出或关闭集群, 关闭时可以通过停止服务或通过 GUI 客户端关闭。退出时应该总是从主节点进行退出或关闭, 其它节点则自动退出。

**注意:** 关闭时应保证其它的数据库服务不再使用此集群服务。

### 1.2.4 卸载集群服务

卸载集群时请先关闭集群服务若无需保留数据文件, 删除整个目录即可。若需要保留数据文件, 请备份或移出 node.toml 和 tdb 目录。然后移除整个目录。

## 1.3 时序数据库

### 1.3.1 安装时序数据库服务

下载或购买数蚕时序数据库程序包, 解压到需要的安装位置中。注意数蚕时序数据库的默认的数据目录为程序目录中的 tdb 目录。因此确保程序目录中有足够的扩展空间, 以免影响未来的使用。

完整的系统目录和文件作用如下:

文件名	说明
doc	数蚕时序数据库的技术参考文档 每个文档以语言缩写区分
cluster	集群服务安装文件
sdk	数蚕时序数据库的开发sdk 分别包含c、java、和python开发包

每个开发语言有一个sample的基础使用示例

sct_tsdb_c	数蚕时序数据库的命令行客户端 使用方法: sct_tsdb_c 用户名 密码 ip地址 端口 详细的命令行使用请使用help命令进行查看
sct_tsdb_dis_s	数蚕时序数据库集群版的服务端执行程序
sct_tdb.toml	数蚕时序数据库配置文件
gen_pwd	密码生成执行程序

其中 sct\_tdb.toml 是数蚕时序数据库配置文件, 配置文件是一个 toml 文件。toml 是标准的 toml 文件规范实现, 详细的 toml 格式和说明请参考 toml 规范说明。程序目录下 sct\_tdb.toml.sample 是一个最小的配置文件示例。

一个标准的数据库配置至少包含 server 表中 port(端口), cluster\_ip(集群 ip), cluster\_port(集群端口) 另外为了保障数据库验证 users 表中应至少含有一个有效用户。

默认密码信息如下:

用户名	密码
root	root
user	user

更多的配置和说明请参考下一章节配置部分内容。

### 1.3.2 运行时序数据库服务

运行时序数据库集群版时确保**集群服务**运行正常。

在 windows 平台, 可以直接双击 sct\_tsdb\_dis\_s.exe。

也可以使用 sct\_tsdb\_dis\_s -hide 隐藏运行的窗口。

也可以使用 sct\_tsdb\_dis\_s -as\_service 作为系统服务运行。

默认地可以在管理员权限下使用 Install\_Service.bat 安装为一个自动启动的系统服务, 服务名为”数蚕时序数据库服务”(内部名为 sct\_tsdb\_dis\_s)

在 linux 平台下, 直接运行 sct\_tsdb\_dis\_s 或使用加了参数的 run\_s.sh 脚本运行。

默认的数据库服务启动使用 sct\_tdb.toml 配置信息, 默认的配置服务端口为 55558, 默认系统有两个用户。

正常运行时系统控制台窗口会输出版本和许可信息。

```
set_enable_log(enable_log@0x00007FFD25680044, std_handle@0x00007FFD256800D8) with f = 'T', pid = 14056
Copyright © 2019- All rights reserved Beijing ShuCan Technology Co., Ltd.
This software is the official product of Beijing ShuCan Technology Co., Ltd.'s time series database cluster version server.
Any form of modification, merger, distribution, publication, reauthorization and sales are prohibited.
Grant use rights.
版权 © 2019- 所有权利保留 北京数蚕科技有限公司。
本软件为北京数蚕科技有限公司时序数据库集群服务端正式产品，禁止任何形式的修改、合并、分发、出版发行、再授权及销售行为。
授予 使用权。

sct_tsdbs version: 0.2.2-30@73c74e8 by sct tsdb team, mail: lk@shucantech.com code : crane.
will connect to cluster 127.0.0.1:22301
connect to cluster 127.0.0.1:22301 successful
```

### 时序数据库服务正常运行状态

#### 1.3.3 关闭时序数据库服务

若安装为服务，可通过 GUI 客户端关闭或通过系统关闭 (windows 平台可通过服务管理器。linux 平台通过 systemctl 命令停止相应服务)。若未安装为服务，可以通过 ctrl-c 结束服务。**注意**: 不要通过 windows 任务管理器或 linux kill -9 命令结束服务，相关操作可能造成数据写入硬盘异常。

#### 1.3.4 卸载时序数据库服务

卸载时序数据库服务时请先关闭时序数据库服务若安装为系统服务，使用 UnInstall\_Service.bat 删除默认生成的服务。若需要保留数据文件，请备份或移出 sct\_tdb.toml 和 tab.meta 文件。然后删除整个目录。

## 2 配置说明

### 2.1 集群配置文件

集群配置文件由-c 或-conf 选项指定。 以下是一个完整的配置

```
#serializable by sct ar4toml.
ver = 0
[node_info]
  is_master = true
  name      = 'n0'
  ip       = '127.0.0.1'
  port     = 22301
  auto_join = true
  group_id = 0
  master_ip = '127.0.0.1'
  master_port = 22301
  max_space = 0x0780000000
  res      = []
  workers  = [ { is_master = false, name = 'n1', ip = '127.0.0.1', port = 22302, auto_join = true
                , group_id = 0, master_ip = '127.0.0.1', master_port = 22301, max_space = 0x0780000000 }
                , { is_master = false, name = 'n2', ip = '127.0.0.1', port = 22303, auto_join = true
                , group_id = 0, master_ip = '127.0.0.1', master_port = 22301, max_space = 0x0a00000000 } ]
```

#### 字段及释义

ver	当前节点保证的内部版本，当前应总是为0，字段必须存在
[node_info]	节点配置信息，必须存在
is_master	是否为主节点，对主节点为true，非主节点为false，必须存在
name	当前节点名，字段必须存在
ip	当前节点ip，内部使用，字段可选
port	当前节点服务监听端口，默认的端口为22301，端口不应被其它服务占用 字段必须存在
auto_join	是否自动加入集群，若自动加入则节点启动后自动加入主节点 否则，节点需要手动加入(客户端)，字段必须存在
group_id	组点编号，存储空间编排组号，为0则不成组，否则以组号成组。 注意：编组功能为实验性功能(平衡读写性能)，不建议在正式生产环境中使用。 组号分配应该连续且不冲突，若组号不为0则以相同组号为一组文件操作依次以固定大小操作物理节点
相同组号的数据分配空间大小必须相同，且所有节点数据写入后不能修改 编组操作可以平均读写性能可以满足使用需求时，但会进行较多的网络通信 因此在读写性能可满足使用条件下，编组数量尽量不要超过3个节点	
master_ip	字段必须存在 主节点IP地址，从节点连接使用，字段必须存在

<code>master_port</code>	主节点服务端口，从节点连接使用，字段必须存在
<code>max_space</code>	最大分配空间，分布式文件当前节点使用的最大空间 注意，这个空间是每个文件可以使用的最大的空间 如此节点存储10个文件，则每个文件可以使用的最大空间均不能超过 <code>max_space</code> 因此，设计物理空间时应充分考虑节点的文件数量 字段必须存在
<code>res</code>	节点已经保存的分布式文件信息，此字段信息应该由集群自动处理不应进行手动修改 主节点中此字段必须存在，非主节点可选存在
<code>workers</code>	从节点信息，此字段由集群管理GUI自动处理，也可手动修改节点中信息为一个完整的从节点
<code>node_info</code> 信息	主节点中此字段必须存在，非主节点可选存在

**注意:** `node.toml` 文件集群会根据当前使用文件状态动态更新 `res` 和 `workers` 信息。因此，此文件任何无关的字段信息或注释信息可能会被重写。

集群可以通过 `-c` 或 `-conf` 命令选项指定不同的启动配置文件，但集群关闭时总是保存更新信息为 `node.toml` 文件。

## 2.2 时序数据库配置文件

集群配置文件为 `node.toml` 文件，以下是一个完整的配置

```
[server]
port = 55558 #default port 55558

cluster_ip = "127.0.0.1"
cluster_port = 22301

[users]
users_list = [ { name = "root", password = "EBB7ADFB334DD96ABD182CB560DED2C6" },
               { name = "user", password = "A9E839B7B6661081D38C6093C4CBC280" } ]
```

### 字段及释义

<code>[server]</code>	服务配置信息，必须存在
<code>port</code>	当前节点服务监听端口，默认的端口为55558 端口不应被其它服务占用 字段必须存在
<code>cluster_ip</code>	集群主节点IP地址，字段必须存在
<code>cluster_port</code>	集群主节点服务端口，字段必须存在
<code>[users]</code>	用户配置信息，必须存在
<code>name</code>	用户名
<code>password</code>	用户哈希密码

用户密码可以由 gen\_pwd 生成，gen\_pwd 生成方法为  
gen\_pwd 字符串

## 3 数据操作

### 3.1 对象和概念

数蚤时序数据库假定用户以下场合的数据使用场景。用户需要把一系列包含时间信息的多个数据点信息快速的保存在外部存储设备中，并可能对保存的数据进行部分区间的数据查询和采样。

数蚤时序数据库假定用户操作的是纵横相间的表格，这个表格叫做时序**数据表**。多个不同的表组成一个库，这个库叫做时序**数据库**。

### 3.2 数据表

每个数据表的同一列数据类型相同，并且总有一列指示时间点(时序性)。表的每一列只能为以下的数据类型(对应 c++ 相应的数据类型)。

i08	有符号1字节整形
u08	无符号1字节整形
i16	有符号2字节整形
u16	无符号2字节整形
i32	有符号4字节整形
u32	无符号4字节整形
i64	有符号8字节整形
u64	无符号8字节整形
float	四字节IEEE单精度浮点数
double	八字节IEEE双精度浮点数
string	字符串
date	日期
time	时间
datetime	日期和时间

表中的一行有多列数据，每个数据含有一个时间点，其中时间点支持部分操作符号。每一行数据只能更新数据不能更新时间点，同时追加数据时间点必须不比最新数据时间点旧(时间单调性)。表的行可以无限的追加，更新但不可删除(详见**存储引擎原理说明**)。

一个表对象可以**创建**，**删除**，**清空**，**显示信息**和**进行数据交换**(导入/导出)操作(见**表命令**)。

### 3.3 库操作

一个时序数据库可能存储于外部或本地，可以进行数据交换和管理。因此数据库支持**连接**、**退出**、**备份**、**还原**操作(见**库命令**)。

### 3.4 限制说明

表名限制

数蚤时序数据使用文件保存表因此表名不应使用本地系统不支持的文件名  
见 windows 文件名规范和相关的 linux 文件命名规范。

#### 表数量限制

受限于操作系统最大打开文件要限制，每个系统可能建立的最大表数量取决操作系统设置。

#### 列空间限制

数蚕时序数据支持数据表最大列空间为 262144 字节

既每一列使用空间相加不超过 262144 字节

#### 字符串长度限制

数蚕时序数据库当前字符串数据类型支持最大长度为 256。

#### 连接数量限制

当前数蚕数据库不同系统网络处理模型不同。linux 平台最大同时在线连接数不超过 32。

#### 查询行数限制

为了保障数据查询和命令的误操作，默认的所有查询命令最大输出数据量为 8192。

若查询数据段数据量远大于数据查询输出量，则等距采样输出数据。

若需要不使用此限制可手动指定最大输出数据量 (见 **max** 说明)。

#### 时间点起始限制

数蚕时序计算数据点以 2001-01-01 00:00:00 起始，结束于 2585-07-21 23:34:33.7096

#### 相对时间分辨率限制

数蚕时序数据库提供基于服务器系统 API, windows 平台的小于不低于 1 微秒, linux 平台不低于 100 纳秒分辨率的单调性时间输出。

#### 集群分配空间最小限制

集群节点空间使用要求每个节点最小分配空间不小于 4G。

## 4 命令规范

命令解析处理流程每个输入串总是先以空格分割, 然后依据命令语法解析执行。每个输入的参数只能为整数、浮点数、字符串、日期、时间及日期时间格式。

### 整数

每个输入参数串转换到对应的数值。

整形数据类型支持十进制, 十六进制, 和二进制格式以下有有效的整数输入 (//后为解释说明)

0	//有符号或无符号整数
1	//省略符号整数
100	//省略符号整数
+1	//有符号整数
-100	//有符号整数
-1000	//有符号整数
+0b000	//有符号二进制整数
-0xa0b3	//有符号十六进制整数
0xA0cD	//无符号大小写混写十六进制整数

**注意:** 数据转换时会根据数据类型依次计算, 对于数据过长的数据可能会造成数据转换溢出。

### 浮点数

以下有有效的浮点数输入 (//后为解释说明)

+0.	//无小数部分正浮点数
-0.01	//仅小数部分负浮点数
-100.0	//仅整数部分负浮点数
-100e20	//仅整数及正指数部分负浮点数
1.25E-32	//小数及负指数部分正浮点数
1.25E+20	//小数及正指数部分正浮点数

### 字符串

字符串则以双引号”开始和结束, \x, \u, \U 转义

以下为有效的字符串 (//后为解释说明)

""	//空串
"abc"	//英文
"中文"	//中文字符
"\x00\x11\x22"	//\x转义
"abc\u1323\Uaabbccdd"	//\u, \U转义

### 日期

日期使用-分割, 必须指定完整的年月日

以下为有效的日期输入

2022-05-21

**时间日期使用:** 分割, 必须指定完整的时分秒, 和可选的纳秒计数

以下为有效的日期输入

12:22:22

12:22:22.12  
12:22:22.123431

### 日期时间

日期时间日期部分须符合日期规范，时间部分须符合时间规范，中间以一个空格分割

以下为有效的时间输入

2022-05-21 12:22:22.123431

## 4.1 数据交换格式规范

数蚕时序数据库使用二进制或命令文件保存原生的数据库或表信息。其中二进制为原始的数据内容，命令文件为对应的执行命令文件。包含多行的命令文件总是以 \n 为分割符。

对外的交换数据，使用 csv 文件或 csvs 文件格式。其中 csv 文件是以分割符 (默认为,) 进行分割的交换格式。数蚕规范的 csv 文件应该首行包含列名信息，行尾以 \n 结束。且相应的数据点中不存在分割符。若存在分割符号，可以使用 csvs 文件保存，csvs 文件是把字符串数据转换为对应十六进制文本。其它的规范同 csv 文件。

## 4.2 命令规范

以下命令中空格表示可以任何多个的空格分割, 中括号表示可选的参数, 斜体部分表示命令参数。

### 4.2.1 库命令

库管理使用 backup 进行备份, restore 恢复。

#### 备份命令

backup as bin *file\_path* [replace]

备份数据库为二进制格式文件到 *file\_path*  
若指定 replace 参数，则自动替换对应文件。

backup as cmd *file\_path* [replace]

备份数据库为命令格式文件到 *file\_path*  
若指定 replace 参数，则自动替换对应文件。

#### 恢复命令

restore from bin *file\_path*

从二进制格式文件 *file\_path* 恢复数据库

restore from cmd *file\_path*

从命令格式文件 *file\_path* 恢复数据库

以下为库操作命令示例

```
backup as bin a.sbk [replace] //备份数据库为二进制格式文件 a.sbk
backup as cmd a.sbk          //备份数据库为命令格式文件 a.sbk
restore from bin a.sbk       //从二进制格式文件a.sbk恢复数据库
restore from cmd a.sbk       //从命令格式文件a.sbk恢复数据库
```

## 4.2.2 表命令

### 创建表命令

`create table_name f1 t1 f2 t2 ....` 创建含有 f1, f2... 字段的表。其中 t1, t2 必须为以下之一

<code>i08</code>	//有符号1字节整形
<code>u08</code>	//无符号1字节整形
<code>i16</code>	//有符号2字节整形
<code>u16</code>	//无符号2字节整形
<code>i32</code>	//有符号4字节整形
<code>u32</code>	//无符号4字节整形
<code>i64</code>	//有符号8字节整形
<code>u64</code>	//无符号8字节整形
<code>float</code>	//四字节IEEE单精度浮点数
<code>double</code>	//八字节IEEE双精度浮点数
<code>string</code>	//字符串
<code>date</code>	//日期
<code>time</code>	//时间
<code>datetime</code>	//日期和时间

### 删除表命令

`drop table_name`

删除表 `table_name`

### 清空表命令

`truncate table_name`

清空表 `table_name`, 但保留表结构

### 保存表命令

`save table_name as bin path [replace]`

以二进制格式保存 `table_name` 到 `path`, 若指定 `replace` 参数, 则自动替换对应文件。

`save table_name as cmd path [replace]`

以命令格式保存 `table_name` 到 `path`, 若指定 `replace` 参数, 则自动替换对应文件。

### 加载表命令

`load table_name from bin path`

以二进制格式加载 `path` 到 `table_name` 表中。

`load table_name from cmd path`

以命令格式加载 `path` 到 `table_name` 表中。

### 导出表命令

`export table_name as csv path [split by ,]`

以 csv 格式导出 `table_name` 到 `path`, 若指定 `split by` 参数, 则使用对应分割符号。

`export table_name as csvs path [split by ,]`

以 csvs 格式保存 `table_name` 到 `path`, 若指定 `split by` 参数, 则使用对应分割符号。

### 导入表命令

```
import table_name from csv path [split by ,]
```

以 csv 格式从 path 导入到 table\_name 中，若指定 split by 参数，则使用对应分割符号。

```
import table_name from csvs path [split by ,]
```

以 csvs 格式从 path 导入到 table\_name 中，若指定 split by 参数，则使用对应分割符号。

**注意:** import 时表结构必须存在，load 会清空原始数据，import 则追加新数据。

#### 显示表命令

```
show tables
```

显示当前库中所有数据表信息

#### 字段信息命令

```
field_of table_name
```

显示表 table\_name 中字段信息

#### 导出部分数据命令

```
export_data tn f1 f2 ... [btw 时间点 1 时间点 2] [max count] as csv path [split by , [replace]]
```

以 csv 格式导出 tn 表中 f1.. 字段在时间点 1 和时间 2 范围内数据到路径 path

若指定 replace 参数，则自动替换对应文件。

其中时间点可以为绝对时间点和相对时间点 (见[时间点规范](#))。

以下为有效的表操作命令示例，//后为解释说明

```
create tab1 f1 u08 f2 u32 //创建表
drop tab1 //删除表
truncate tab1 //清空表
show tables //显示表
field_of tab1 //显示表字段信息
load tab from bin abc.bin [replace] //从bin文件加载表
load tab from cmd abc.tcd [replace] //从cmd文件加载表
import tab from csv abc.csv [split by ,] //从csv文件导入表
import tab from csvs abc.csvs [split by ,] //从csvs文件导入表
save tab as bin abc.bin [replace] //保存表到bin文件
save tab as cmd abc.tcd [replace] //保存表到cmd文件
export tab as csv abc.csv [split by ,] //导出表到csv文件
export tab as csvs abc.csvs [split by ,] //导出表到csvs文件
export_data tn f1 f2 btw -1h +1h as csv a.csv //导出表中当前前后一小时数据到a.csv
```

### 4.2.3 数据写操作

#### 写操作命令

```
push into tn * values v1 v2
```

向 tn 中在当前时间点以字典序字段追加值为 v1, v2... 的数据

```
push into tn f1 f2 ... values v1 v2
```

向 tn 中在当前时间点追加 f1, f2... 值为 v1, v2... 的数据

```
push into tn f1 f2 ... values v1 v2 at 时间点
```

向 tn 中在时间点追加 f1, f2... 值为 v1, v2... 的数据

`push into tn f1 f2 ... values v1 v2 at 相对时间`

向 `tn` 中在相对时间追加 `f1, f2...` 值为 `v1, v2...` 的数据  
其中时间点可以为绝对时间点和相对时间点 (见[时间点规范](#))。

以下为有效的数据写操作命令示例, `//`后为解释说明

```
push into t * values 12 //以字典序追加在当前时间点
push into t id values 12 //默认追加在当前时间点
push into t id values 12 at 2022-05-21 12:22:22.123431 //追加在时间点
push into t id values 12 at >10s //追加在相对开始时间点10s
```

#### 4.2.4 数据更新操作

##### 更新命令

`update tn set f1 v1 f2 v2 ... at 时间点`

更新 `tn` 表中时间点的 `f1, f2` 值为 `v1, v2`

其中时间点可以为绝对时间点和相对时间点 (见[时间点规范](#))。

以下为有效的更新命令示例

```
update tab set id 123 name "kkk" at 2019-11-21 18:22:20
update tab set id 123 name "kkk" at -10s
```

#### 4.2.5 数据读操作

`view tn * [max 8192]`

查看 `tn` 中在所有时间段的数据, 默认最大采样输出 8192 个

`view tn f1 f2 ... btw 时间点 1 时间点 2 [max 8192]`

查看 `tn` 中在 [时间点 1, 时间点 2] 区间段的 `f1, f2, ...` 字段数据, 默认最大采样输出 8192 个  
其中时间点可以为绝对时间点和相对时间点 (见[时间点规范](#))。

##### 聚合操作

聚合操作为在字段名后添加一个@和聚合函数名, 当前聚合函数支持 `max, min, sum, count, avg`.  
分别表示获取数据最大值, 最小值, 求和, 计数, 平均值。

**注意:** 当前聚合只支持一个字段处理, 多个字段的处理输出可能会导致数据处理的`不一致`。

以下为有效的读操作命令示例, `//`后为解释说明

```
view tab1 * max 10
view tab1 id btw 2019-12-12 12:12:33.000 2019-12-12 12:12:50.000
view tab1 id max 1024
view tab1 id btw -10h +10m
view tab1 val@max btw -1h +1h
```

#### 4.2.6 时间点规范

时间点包含绝对时间点和相对时间点两种格式。

绝对时间点格式为标准时间日期格式。

相对时间点操作分别如下定义

操作符号,时间值,时间单位

+ 1000 s

操作符号如下

+表示当前时间加上时间间隔

-表示当前时间减去时间间隔

>表示开始时间加上时间间隔

<表示开始时间减去时间间隔

/表示最后时间加上时间间隔

\表示最后时间减去时间间隔

时间单位如下:

Y年

M月

D日

h时

m分

s秒

ms毫秒

us微秒

ns纳秒

相对时间点计算基于服务器的时间计算

## 5 数据库原理

### 5.1 命令执行原理

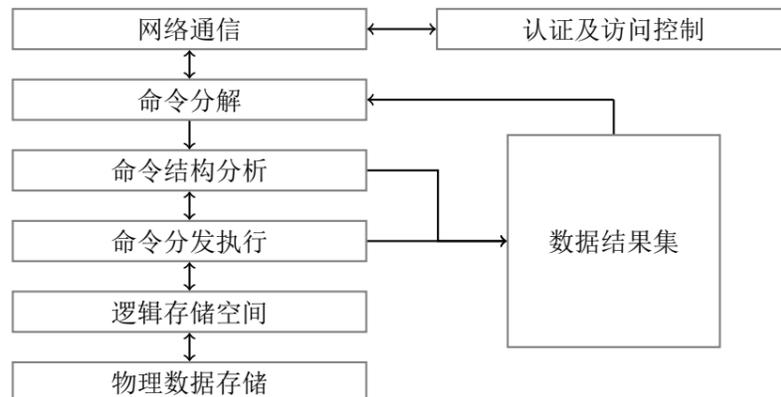


图 1: 时序数据库命令通信流程图

### 5.2 数据存储原理

数蚕数据库支持多种物理存储引擎根据业务需求不同可能会使用不同的存储引擎。针对快写存储场景，当前支持直接追加数据的线性只追存储引擎 (LAE) 和 LSM(基于 LSM-树) 存储引擎。

当前数据库使用的默认引擎为 LAE 引擎。LAE 引擎直接基于文件进行追写操作，数据追加操作为常数 1 复杂度。读取时基于文件系统进行时间点的二分查找，读取复杂度为  $\log N$ 。

LAE 根据不同的参数策略可能会使用缓存数据进行读取或直接的固定点采样相应操作复杂度为数据量常数。

LAE 把每一行数据作为一个整个数据块存储到外部设备中。每个数据块以固定的大小 (每个表不同) 存储。

### 5.3 存储数据块

每个存储块大小取固定的存储大小，最大存储大小为 262144 字节 (含时间点) 以下是一个可能的块存储示意



一个表数据存储多个固定大小数据块，数据总是追加存储。

### 5.4 集群连接原理

数蚕时序数据集群当前使用强一致性模型。其中数据一致性由主节点进行控制。

主节点拥有所有节点和空间分配信息。从节点拥有空间分配信息，但所有数据操作通过主节点进行。

集群启动时各个节点根据是否加入标志连接主节点，主节点根据加入节点提供的空间信息和组信息计算新的空间分配信息。并分发新节点信息到所有其它节点，其它节点调整内部节点空间信息。等待所有配置节点加入集群成功后集群可以响应外部数据操作请求。

集群主节点退出时或异常时，所有从节点会响应退出信息并立即退出。

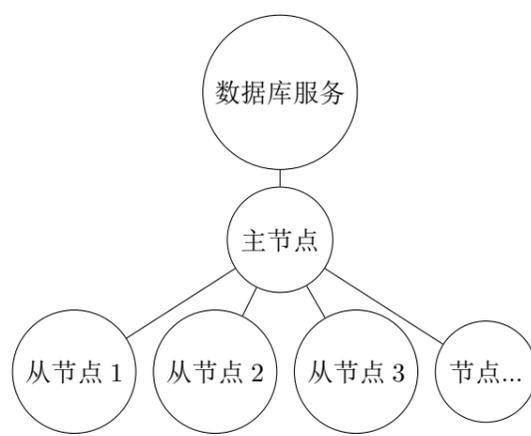
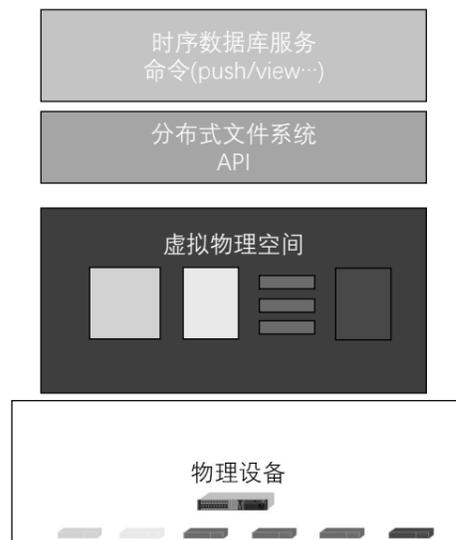


图 2: 集群连接拓扑图

### 5.5 分布式存储原理

数蚕时序数据库假定外部存储为物理文件。所有操作 LAE 引擎直接操作于物理文件。集群系统提供了外部的虚拟分布式文件系统。它通过提供完整一致的文件读写 API(fopen/fread/fwrite...)。对时序数据库提供了用户透明的分布式文件操作。



时序数据库分布存储原理

## 6 开发

### 6.1 开发包安装和部署

安装数蚕时序数据库开发包位于产品安装位置的 sdk 目录中。使用时请把对应安装包放置于开发项目中，或添加相关库的开发路径。

部署开发包部署时根据各语言要求分别附加相应的开发库。

**注意:** 对于单机版的数蚕时序数据库产品开发包，需要附加产品购买的许可证。

### 6.2 基础开发流程

对于网络版或集群版数据库开发 sdk 总是以下的操作流程。

- 1, 建立和时序数据库服务的连接
- 2, 执行操作命令
- 3, 获取执行结果数据集
- 4, 提取或转换数据集结果数据集通过 at 获取对应行列单元值

每个值通过符号，类型和大小判断，通过 as 函数转换为对应的语言数据。

- 5, 断开和数据库服务的连接

对于单机版开发 sdk 则无需 1, 3 步骤。

以下为一个基于 c 语言的基础使用示例

```
#include <stdlib.h>
#include <stdio.h>
#include "tsdb_lib.h"

void print_bymyself(void* d) {

    int r = rows_of(d);
    int c = cols_of(d);
    printf("\n");
    for (size_t i = 0; i < c; ++i) {
        int s = 0;
        const char* n = col_name(&s, d, i);
        printf("%.*s ", s, n);
    }
    printf("\n");
    for (size_t i = 0; i < r; ++i) {
        for (size_t j = 0; j < c; ++j) {
            void* v = at(i, j, d);
            int g = val_sign(v); int s = val_size(v); int t = val_type(v);
            if (t == VAL_TYPE_I) {
                if (s == VAL_SIZE_1) {
                    printf(" %d", g == VAL_SIGN ? as_int8(v) : as_uint8(v));
                } else if (s == VAL_SIZE_2) {
```

```

        printf(" %d", g == VAL_SIGN ? as_int16(v) : as_uint16(v));
    } else if (s == VAL_SIZE_4) {
        printf(" %d", g == VAL_SIGN ? as_int32(v) : as_uint32(v));
    } else {
        printf(" %d", g == VAL_SIGN ? as_int64(v) : as_uint64(v));    }
} else if (t == VAL_TYPE_F) {
    printf(" %f", s == VAL_SIZE_4 ? as_float(v) : as_double(v));
} else if (t == VAL_TYPE_D) {
    int y, m, d; as_date(&y, &m, &d, v);
} else if (t == VAL_TYPE_S){
    char* p = NULL; int c = 0;
    as_str(&p, &c, v);
    if( p != NULL ){
        printf(" %.*s", c, p); free(p);
    }
} else {}
}
printf("\n");
}
}

int main(int argc, char** argv) {
    if(!connect_to_server("127.0.0.1", 55558, "user", "password")){ return -1; }

    void* d = exec("view tab id value max 10");
    if( d == NULL ){ printf("execute tsdb command failed\n"); return 0; }
    print(d);
    print_bymyself(d);
    del_datas(d);
    exit_to_server();
    return 0;
}

```

## 6.3 语言相关

### 6.3.1 c

数蚕时序数据库使用标准 c 开发接口。

其中 4zh.h 为中文的注释说明, \_local 后缀为单机版 sdk。

其中 windows 平台基于 vs2019 专业版 x64 编译动态库, 位于 sdk/c/dll 目录中

linux 平台提供系统默认运行环境的 g++编译 x64 动态库, 位于 sdk/c/so 目录中

头文件位于 sdk/c/inc 目录。

tsdb\_lib\_sample 是一个基础的开发示例。

### 6.3.2 c++

数蚕时序数据库使用标准 c 开发接口。c++开发使用同样开发库。

### 6.3.3 java

数蚕时序数据库 java 语言 sdk 提供基于 jdk1.8.0\_202 版本编译 jar 包。目录中 xxx\_test.java 是一个完整的对应上述流程的基础的开发示例。

### 6.3.4 python

数蚕时序数据库 python 语言 sdk 提供基于 python2.7.18 版本的编译库 tsdb\_lib.pyc。以及 3.10.1 版本的编译库文件 tsdb\_lib\_py3.pyc。tsdb\_lib\_sample.py 是一个完整的对应上述流程的 python2 的开发示例。tsdb\_lib\_sample\_py3.py 是一个完整的对应上述流程的 python3 的开发示例。

### 6.3.5 其它语言或版本

目前暂不提供其它语言支持。若有特殊语言需求，请联系相关销售或邮件到 [admin@shucantech.com](mailto:admin@shucantech.com) 反馈相关需求。

## 6.4 常见问题

1，数据库服务无法正常启动数蚕时序数据库默认使用 tcp 的 55558 端口。请确保连接的服务器服务 55558 端口未被占用。

2，数据库服务无法正常连接数蚕时序数据库默认使用 tcp 的 55558 端口。请确保连接的服务器服务运行正常，防火墙开放 55558 端口。数据库为防止暴力猜解密码默认的最大连接次数为 30，超过重试错误次数，1 分钟内禁止连接。1 分钟后最大尝试次数为 20，超过重试错误次数，30 分钟内禁止连接。30 分钟后最大尝试次数为 5，超过重试错误次数，15 天内禁止连接。15 天后最大尝试次数为 3，超过重试错误次数，永久禁止连接。直至服务重启。

3，数据库服务数据操作总是失败数蚕时序数据库需要连接到集群存储服务器，未正常连接时所有数据操作不响应。因此请确保数蚕时序集群服务正常启动，并可正常访问对应主节点端口。

## 7 维护

数蚕时序数据库设计上可以长驻运行，对于使用空间和运行异常仍然可能需要进行必要的维护。

### 7.1 集群管理

集群是分布式数据存储的基础，因此应该保障集群运行时无故障存在。集群可以通过使用 GUI 客户端连接主节点进行日常的监督和查看。GUI 客户端可以自动或手动更新当前节点状态，对于有离线状态的节点，应确保数据无损坏条件下加入集群。

集群数据文件使用时最大空间为单个文件使用空间，不对整体使用空间进行约束，因此在集群使用较多文件资源时应该关注物理使用空间是否充分。及时清理调整当前节点使用空间。

集群节点最后节点或组是数据最后有效操作空间，当数据空间不足时应该注意扩展物理节点保障存储空间。

集群应该按照**集群服务**中操作规范进行正常启动和关闭。

#### 7.1.1 集群备份和扩展

集群资源数据正常操作时无需干预处理，数据备份操作由上层数据服务进行。若需要手动进行物理分布式文件备份，应在所有集群节点正常关闭下，备份所有节点中 toml 配置文件和 tdb 资源文件。

集群扩展集群资源需要扩展时，应通过 GUI 客户端添加新的节点并加入集群中。集群扩展操作时停止上层数据服务，避免数据操作可能出现异常。

#### 7.1.2 集群异常处理

集群节点异常退出时可能造成数据的不一致。安装目录下.log 会记录少量的出错信息，相关信息请专业人员进行处理。

### 7.2 时序数据库管理

时序数据库日常运行应该按照**时序数据库**中操作规范进行正常启动和关闭。

### 7.3 时序数据库备份和恢复

时序数据库应该根据使用需要使用 backup 进行备份和 restore 进行恢复。注意，数据备份和恢复时使用数据在数据库服务器，因此数据库服务器机器应保留充分的物理空间。避免数据备份空间不足导致备份或恢复失败。

### 7.4 时序数据库异常处理

时序数据库异常退出时可能造成数据表信息不正常写入。安装目录下.log 会记录少量的出错信息，相关信息请专业人员进行处理。

## 7.5 产品更新和升级

数蚕时序数据库产品正常运行时会自动检查产品更新信息。相应服务运行窗口显示产品更新信息。若需要进行更新或升级请联系产品销售人员或在官方网站查找对应产品。

注意，不同的产品许可证在服务期外可能不能用于新的产品。

## 7.6 商业技术支持或服务

更多的关于数蚕时序数据的技术支持或服务请联系产品销售人员或在网站产品页商务咨询中留言或邮件 [admin@shucantech.com](mailto:admin@shucantech.com) 反馈洽谈。