ALIBABA CLOUD

阿里云

时序时空数据库

时序数据库 InfluxDB® 版

文档版本：20220704

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.InfluxDB®介绍

时序数据库 InfluxDB®版是一款专门处理高写入和查询负载的时序数据库，用于存储大规模的时序数据并进行实时分析，包括来自DevOps监控、应用指标和IoT传感器上的数据。

主要特点

InfluxDB®是您处理时序数据的一个绝佳选择，目前有以下特点：

- 专为时间序列数据量身打造的高性能数据存储。TSM引擎提供数据高速读写和压缩等功能。
- 简单高效的HTTP API写入和查询接口。
- 针对时序数据，量身打造类似SQL的查询语言，轻松查询聚合数据。
- 允许对tag建索引，实现快速有效的查询。
- 数据保留策略（Retention policies）能够有效地使旧数据自动失效。

2.实例规格和产品计费

本文介绍云时序数据库InfluxDB®的主实例规格，帮助您了解云时序数据库InfluxDB®主实例的最新规格信息和历史规格信息。

存储价格

云时序数据库InfluxDB®目前提供三种存储类型：SSD云盘、高效云盘和ESSD PL1云盘。存储类型的说明如下表所示。

云盘类别	说明
SSD云盘（推荐）	SSD云盘是指基于分布式存储架构的SSD弹性块存储设备，具有低时延、高性能、持久性、高可靠等性能，支持随时创建、扩容以及释放。
高效云盘	高效云盘是指基于分布式存储架构的一般弹性块存储设备，性能和吞吐略低于SSD云盘，具有持久性、高可靠等性能，支持随时创建、扩容以及释放。
ESSD PL1云盘（推荐）	增强型（Enhanced）SSD云盘，是阿里云全新推出的超高性能云盘产品。ESSD云盘基于新一代分布式块存储架构，结合25GE网络和RDMA技术，提供更高的随机读写能力和更低的单路时延能力。

性能级别的介绍请参见块存储性能，云InfluxDB®实例的吞吐也与写入请求的Batch大小相关，具体价格请见下表。

单位：元/GB/月

存储类别	高效云盘	SSD云盘
日本（东京）	0.36	1.5
印度（孟买）	0.34	1.21
新加坡	0.35	1.27
澳大利亚（悉尼）	0.37	1.56
马来西亚（吉隆坡）	0.34	1.21
印度尼西亚（雅加达）	0.35	1.27
中国内地	0.35	1
中国（香港）	0.35	1.27
德国（法兰克福）	0.35	1.27
英国（伦敦）	0.34	1.38
阿联酋（迪拜）	0.42	1.66
美东（弗吉尼亚）	0.35	1.27
美西（硅谷）	0.35	0.35

存储类别	高效云盘	SSD云盘

实例规格和价格

特别注意：

- **以下规格列表中标注的性能指标是基于特定测试案例下的经验数据参考值，仅供购买参考，系统性能数据依赖于具体场景、系统负载、数据结构以及查询复杂度。**
- **参考写性能测试基准：单个写请求包含500个数据点，所有数据只写入一个数据库，一共写入5万时间线，每个时间线包含5对tagkv，以及5个field。**
- **参考读性能测试基准：每次查询扫描10个时间线，1万个原始数据点。**

参考指标说明

- 每秒写入请求指的是每秒写入请求的数量。
- 写入数据点为每秒写入的数据点数量。
- 查询性能为每秒可执行的查询次数，每次查询扫描10个时间线，1万个原始数据点。

单位：元/GB/月

系列		基础版 （高效云盘）						高可用版 （SSD云盘）					
规格		2核8GB	4核16GB	8核32GB	16核64GB	32核128GB	64核256GB	2核8GB	4核16GB	8核32GB	16核64GB	32核128GB	64核256GB
每秒写入请求		~50	~100	~160	~300	~500	~800	~70	~160	~240	~400	~600	~1000
写入数据点		~25000	~50000	~80000	~150000	~250000	~400000	~35000	~80000	~120000	~200000	~300000	~500000
每秒查询请求		~25	~50	~80	~150	~250	~400	~75	~170	~300	~480	~780	~1250
中国	中国内地	274	541	1074	2140	4271	8535	762	1503	2986	5951	11881	23742
	中国（香港）	401	802	1604	3208	6415	12830	1116	2230	4462	8922	17844	35689

系列		基础版 （高效云盘）						高可用版 （SSD云盘）					
亚太	日本（东京）	444	888	1776	3550	7100	14200	1234	2470	4939	9875	19750	39502
	印度（孟买）	319	638	1275	2549	5099	10197	887	1774	1275	7091	14184	28364
	新加坡	401	802	1604	3208	6415	12830	1116	2231	1604	8922	17845	35689
	澳大利亚（悉尼）	407	815	1628	3256	6510	13019	1131	2265	4527	9056	18108	36216
	马来西亚（吉隆坡）	381	762	1524	3047	6094	12187	1059	2120	4239	8474	16951	33902
	印度尼西亚（雅加达）	402	802	1605	3208	6415	12830	1117	2230	4463	8922	17844	35690

系列		基础版 （高效云盘）						高可用版 （SSD云盘）					
欧洲	德国（法兰克福）	424	847	1693	3385	6770	13540	1178	2355	4709	9417	18833	37666
	英国（伦敦）	403	806	1611	3220	6440	12879	1120	2240	4480	8957	17914	35827
美洲	美东（弗吉尼亚）	320	640	1279	2558	5115	10229	890	1779	3557	7114	14228	28455
	美西（硅谷）	412	824	1647	3294	6588	13176	1146	2291	4582	9163	18326	36652

说明 高可用版采用三副本，购买页面容量大小为单节点SSD云盘的容量，实例实际分配容量为购买页容量3倍，所以计算价格为3.0 元/月/GB。

3.购买流程

本文介绍购买InfluxDB®版数据库流程。

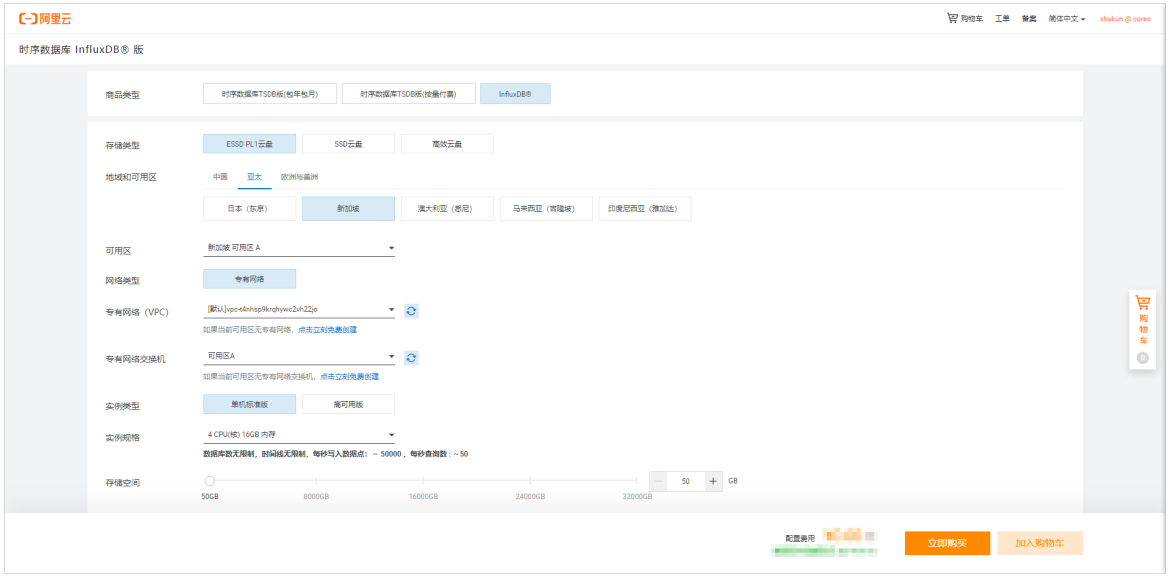
时序数据库TSDB当前不支持创建TSDB实例，建议您登录云原生多模数据库Lindorm控制台创建Lindorm实例。具体操作，请参见创建实例。

前提条件

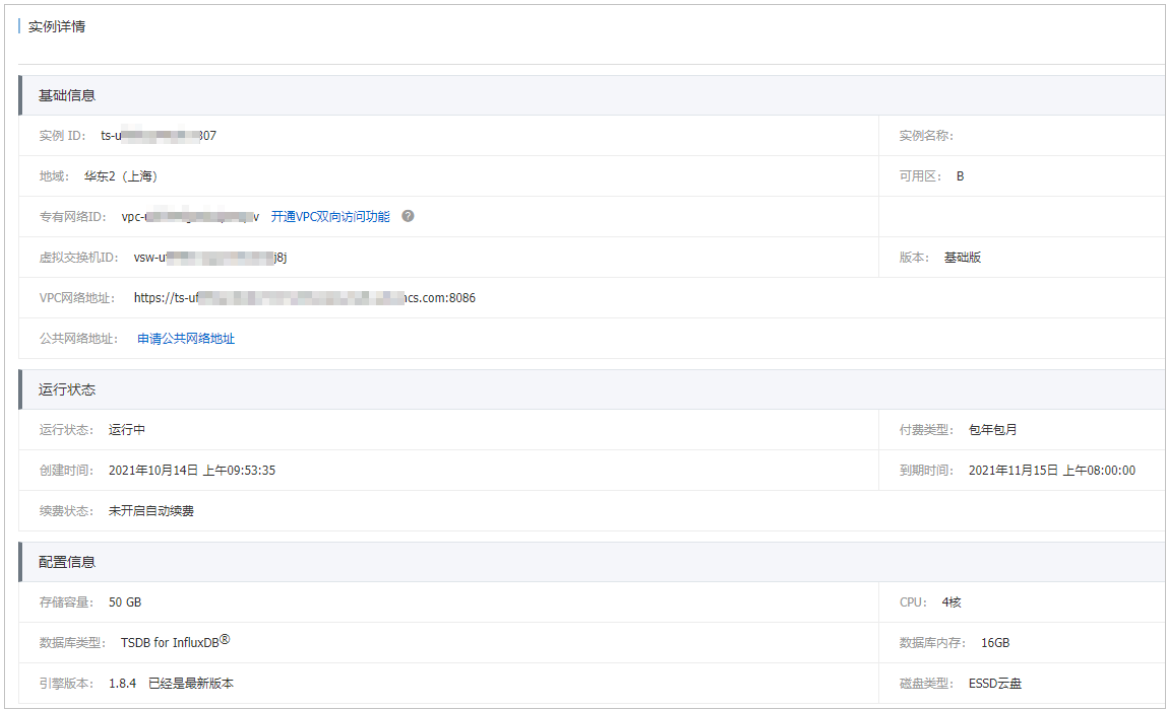
- 您已经注册了阿里云账号并完成实名认证。
- 您已拥有阿里云专有网络（VPC）。关于创建VPC的具体信息，请参见创建专有网络。

购买TSDB For InfluxDB®实例流程

1. 登录阿里云官网。
2. 在顶部导航栏中选择**产品 > 数据库 > 时序数据库TSDB版**，进入TSDB产品主页。
3. 单击**创建**，进入购买页面，选择**InfluxDB®**。然后，选择您所需要的基本配置和购买量，如下所示（下图中的配置仅供参考）。



   - 基本配置
     - 实例规格：最大支持创建10个数据库。
     - 存储空间：最小支持50GB存储容量。
     - 实例名称：长度不能超过128个字符，以大小写字母或中文开头，实例名称可包含数字、‘.’、‘_’或‘-’。
   - 购买量
     - 购买数量：只能选择1，每次只能购买一个实例。
     - 购买时长：从1个月到3年，可选择自动续费。
4. 确认信息无误后，单击**立即购买**。
5. 购买成功并创建实例成功后，进入TSDB主页上的**管理控制台**可以看到关于新建实例的信息。


实例详情
基础信息
实例 ID： ts-u[redacted]07
实例名称：
地域： 华东2（上海）
可用区： B
专有网络ID： vpc-[redacted]v 开通VPC双向访问功能
虚拟交换机ID： vsw-u[redacted]j8j
版本： 基础版
VPC网络地址： https://ts-uf[redacted]ics.com:8086
公共网络地址： 申请公共网络地址
运行状态
运行状态： 运行中
付费类型： 包年包月
创建时间： 2021年10月14日 上午09:53:35
到期时间： 2021年11月15日 上午08:00:00
续费状态： 未开启自动续费
配置信息
存储容量： 50 GB
CPU： 4核
数据库类型： TSDB for InfluxDB®
数据库内存： 16GB
引擎版本： 1.8.4 已经是最新版本
磁盘类型： ESSD云盘

4.管理用户账号和数据库

本文介绍创建、修改和删除时序数据库和账号以及添加账号权限。

前提条件

已购买实例，具体操作请参见购买流程。

账号类型

时序数据库Influxdb版实例支持两种数据库账号：管理员账号和普通账号。您可以在控制台管理所有账号和数据库，账号拥有的具体权限请参见文末账号权限列表。

账号类型	说明
管理员账号	只能通过控制台创建和管理。一个实例中只能创建一个管理员账号，可以管理所有数据库。不能创建和管理其他账号。拥有实例下所有数据库操作的所有权限。
普通账号	只能通过控制台创建和管理。一个实例可以创建多个普通账号 。不能创建和管理其他账号。需要通过控制台给普通账号授予特定数据库的权限。可以对已授权的数据库进行读写操作

管理用户账号

创建用户账号

1. 在管理控制台中的**账号管理**页面中单击**创建**。

2. 进入**创建**页面后，设置账号和密码。

- 可以创建管理员账号和普通账号。1个InfluxDB实例只允许创建1个管理员账号。
- 数据库账号：由小写字母、数字、下划线组成，以字母开头、字母或数字结尾，最长16个字符
- 密码：大写、小写、数字、特殊字符占三种，长度为8-32位，特殊字符为!@#$%^&*()_+-=

创建账号成功后，如下图所示（假设我们在这里创建了一个名为 `db_test` 的账号）。

删除用户账号

在**账号管理**页面中，点击需要删除的用户账号对应的**删除**按钮即可，如下图所示。

管理数据库

创建数据库

1. 在**数据库管理**页面单击**创建**。
2. 进入**创建**页面后，输入数据库名创建数据库，创建成功后，如下图所示（假设我们在这里创建了一个名为 `mydb` 的数据库）。

新创建的数据库默认的保留策略是 `autogen` ，初始化的保留策略时长为0s，表示数据永久保存，如果需要对此进行修改，请单击**存储策略管理**，根据您的需求来修改存储策略，包括创建新的保留策略和修改已有的保留策略。

创建新的保留策略

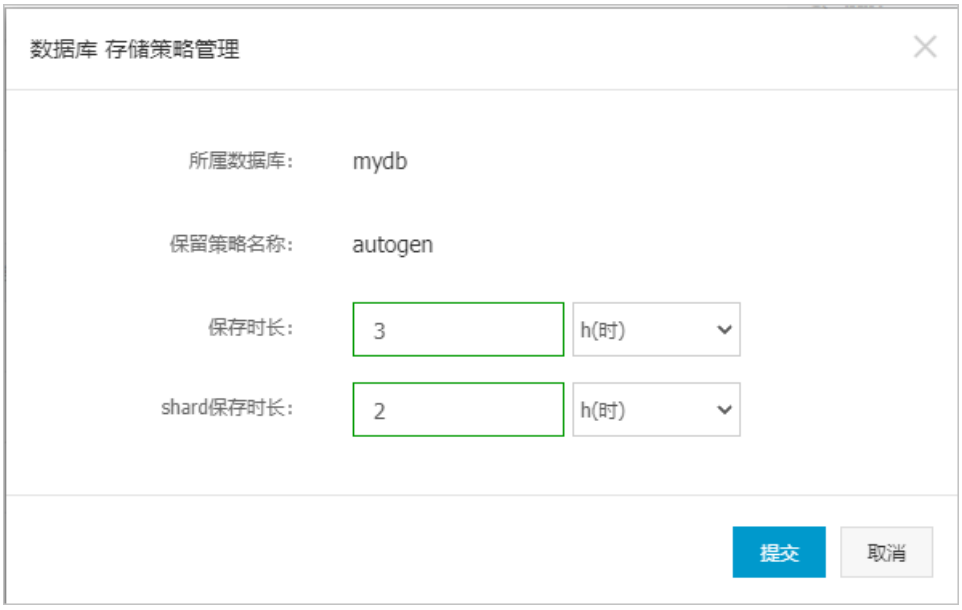
进入到**存储策略管理**页面后，单击**创建**。

创建新的保留策略，在这里，我们创建了名为 `oneday` 的保留策略，保存时长为1天，如下图所示。

修改已有的保留策略

假设我们想要将 `autogen` 的时长设为2小时，那么请单击**修改**。

将时长设为2小时，如下图所示。

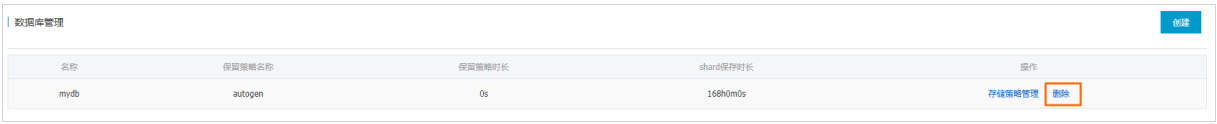


设置默认的保留策略

保留策略描述了数据在TSDB For InfluxDB®中保存的时间，如果在读写数据时不明确指定保留策略，则使用默认（default）的保留策略。

删除数据库

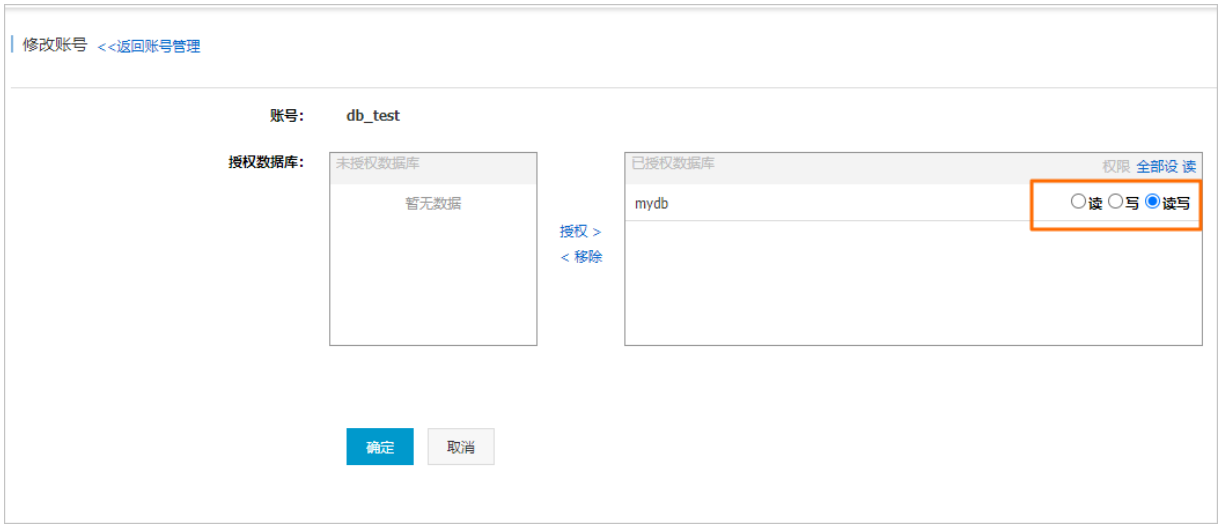
在**数据库管理**页面中，单击需要删除的数据库对应的**删除**按钮即可，如下图所示。



权限管理

用户可修改指定普通账号下数据库的读写权限。

例如，对数据库 `mydb` 授权读写权限，如下图所示。



管理员用户权限

管理员用户具有对所有数据库的完全访问权限。

账号权限列表

账号类型	授权类型	操作对象	权限				
普通账号	读写	数据库	SHOW DATABASES	USE (DATABASE)			
		数据操作	SELECT	INSERT	DELETE (FROM)	CARDINALITY	SHOW TAG KEY (EXACT) CARDINALITY
			SHOW TAG VALUES (EXACT) CARDINALITY	SHOW FIELD KEY (EXACT) CARDINALITY	SHOW MEASUREMENT (EXACT) CARDINALITY	SHOW TAG KEYS	SHOW TAG VALUES
			SHOW FIELD KEYS	SHOW QUERIES	EXPLAIN (ANALYZE)		

		时间序列	SHOW SERIES	DROP SERIES	DROP SERIES		
		连续查询	CREATE CONTINUOUS QUERY	DROP CONTINUOUS QUERY	SHOW CONTINUOUS QUERIES		
		度量	SHOW MEASUREMENTS	SHOW MEASUREMENT (EXACT) CARDINALITY			
		保留策略	SHOW RETENTION POLICIES	DROP RETENTION POLICY			

5.数据迁移

数据迁移功能支持用户将数据从自建的InfluxDB实例迁移到阿里云TSDB for InfluxDB®实例。

前提条件

- 已购买TSDB for InfluxDB®实例。具体操作，请参见购买流程。
- 自建的InfluxDB实例需要满足以下要求：
  - 已开启备份恢复功能。
  - 备份恢复服务端口支持访问。
  - 服务版本需大于等于1.5。

操作步骤

1. 单击目标实例操作列**管理**。
2. 单击左侧**时序数据管理 > 数据迁移**，然后单击**添加迁移任务**。
3. 在配置向导页面，配置以下参数。

参数	说明
任务名称	填写任务名称。支持大写字母、小写字母与下划线。
实例类型选择	选择需要迁移的实例类型。
主机名或者IP地址	数据源所在的主机名或IP地址。仅实例类型选择**用户自建实例**，出现此参数。 **说明** 当前数据迁移服务**仅支持公网地址**，暂时不支持VPC内的地址。
数据访问端口	数据源端口。仅实例类型选择**用户自建实例**，出现此参数。 **说明** 在自建InfluxDB的配置文件中，可在[http]部分查看配置项 `bind-address` 中指定的端口，该端口即数据访问端口。

参数	说明
数据备份端口	数据源端口。仅实例类型选择**用户自建实例**，出现此参数。 **说明** 在自建InfluxDB的配置文件中，可查看全局配置项 `bind-address` 中指定的端口。该配置项的默认值是“127.0.0.1:8088”，无法被外网访问，为保障迁移任务成功执行，需要将 `bind-address` 设置为允许外网访问的地址，例如将其设置为“:8088”，那么此时数据备份端口就是“8088”。配置文件修改后，需要重启InfluxDB服务，配置才生效。为了数据安全性，请在迁移完成后将该设置恢复或者注释掉。
是否开启HTTPS	选择是否开启HTTPS。仅实例类型选择**用户自建实例**，出现此参数。

4. 单击**授权实例访问并进入下一步**，选择需要迁移的数据库。
5. 单击右下角**预检查**，待检查完成后，单击**启动**，开始迁移任务。

6.产品系列

6.1. 产品系列概述

云时序数据库InfluxDB®的实例包括两个系列：基础版和高可用版。本文介绍如何查看产品系列及各系列的差别。

查看产品系列

您可以在**实例详情**页面的**基础信息**栏查看实例所属的系列。

各产品系列对比

系列	说明	适用场景
基础版	单节点实例	个人学习、中小规模DevOps监控、应用指标和IoT传感器的时序数据采集分析、中小企业开发测试。
高可用版	采用Raft一致性协议的三节点架构，适合80%以上的用户场景	大中型企业的指标采集、业务监控（物联网、容器、工业生产等重要链路资源监控）。

> **说明**
>
> 关于各个系列支持的实例规格，请参见实例规格。

6.2. 基础版

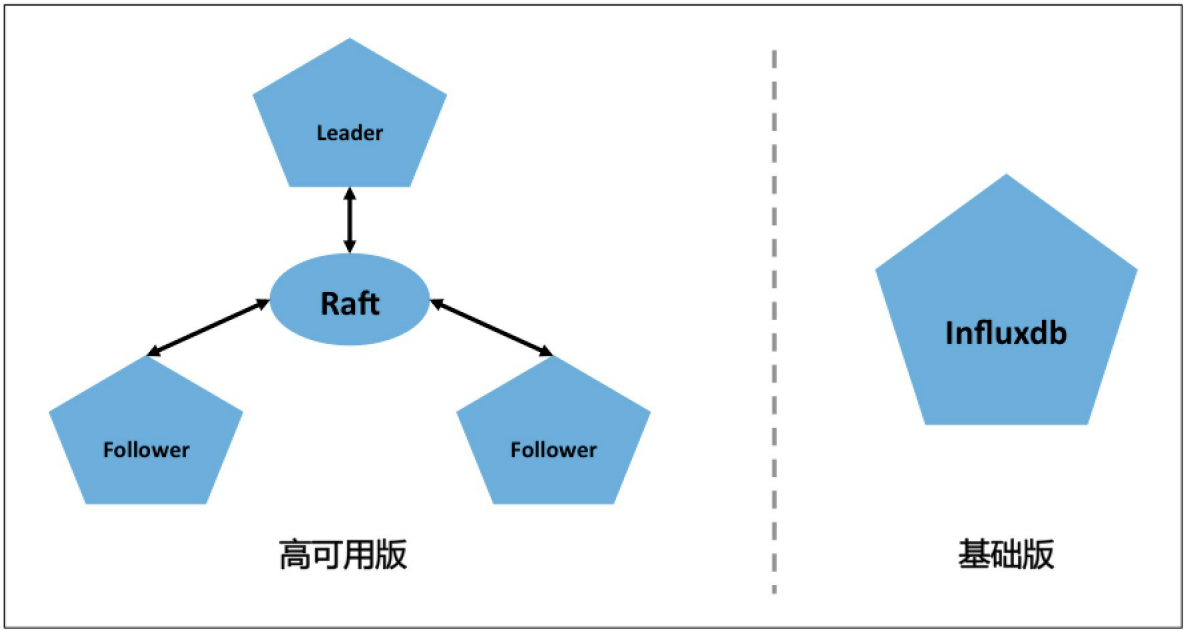
本文档主要介绍了云时序数据库InfluxDB®基础版的相关信息。

简介

基础版称为单机版，只有单个数据节点，采用云盘保证数据可靠，性价比高。

> **说明**
>
> 由于基础版只有一个数据节点，没有备节点作为热备份，因此当该节点意外宕机或者执行变更配置、版本升级等任务时，会出现一段时间的不可用。如果业务对数据库的可用性要求较高，不建议使用基础版，可选择其他系列（如高可用版）。

基础版和高可用版的对比拓扑图如下所示：



功能

基础版支持IP白名单、监控、数据迁移等基本功能：

- 管理用户账号和数据库
- 数据迁移
- 云实例监控

适用场景

- 互联网基础资源监控

- 容器监控
- 业务运营监控分析
- 物联网设备远程实时监控
- 工业安全生产监控
- 生产质量评估和故障回溯

开始使用

您可以通过最佳实践快速入门，向云时序数据库InfluxDB®中写入和查询数据。

- 使用Influx CLI（命令行界面）连接TSDB For InfluxDB®
- 从采集到分析-TSDB For InfluxDB®让你的数据产生价值
- Grafana接入阿里云时序数据库TSDB For InfluxDB®©服务
- Prometheus对接阿里云TSDB For InfluxDB®©服务
- Grafana监控可视化

常见问题

- 云时序数据库InfluxDB®同开源本的InfluxDB有哪些不同？

  详见阿里云文档说明：TSDB for InfluxDB®©与自建InfluxDB对比优势

- 申请了一个实例，连接需要用户名密码，如何设置？

  详见阿里云帮助文档：管理用户账号和数据库

- 云时序数据库采用HTTPS加密传输吗？

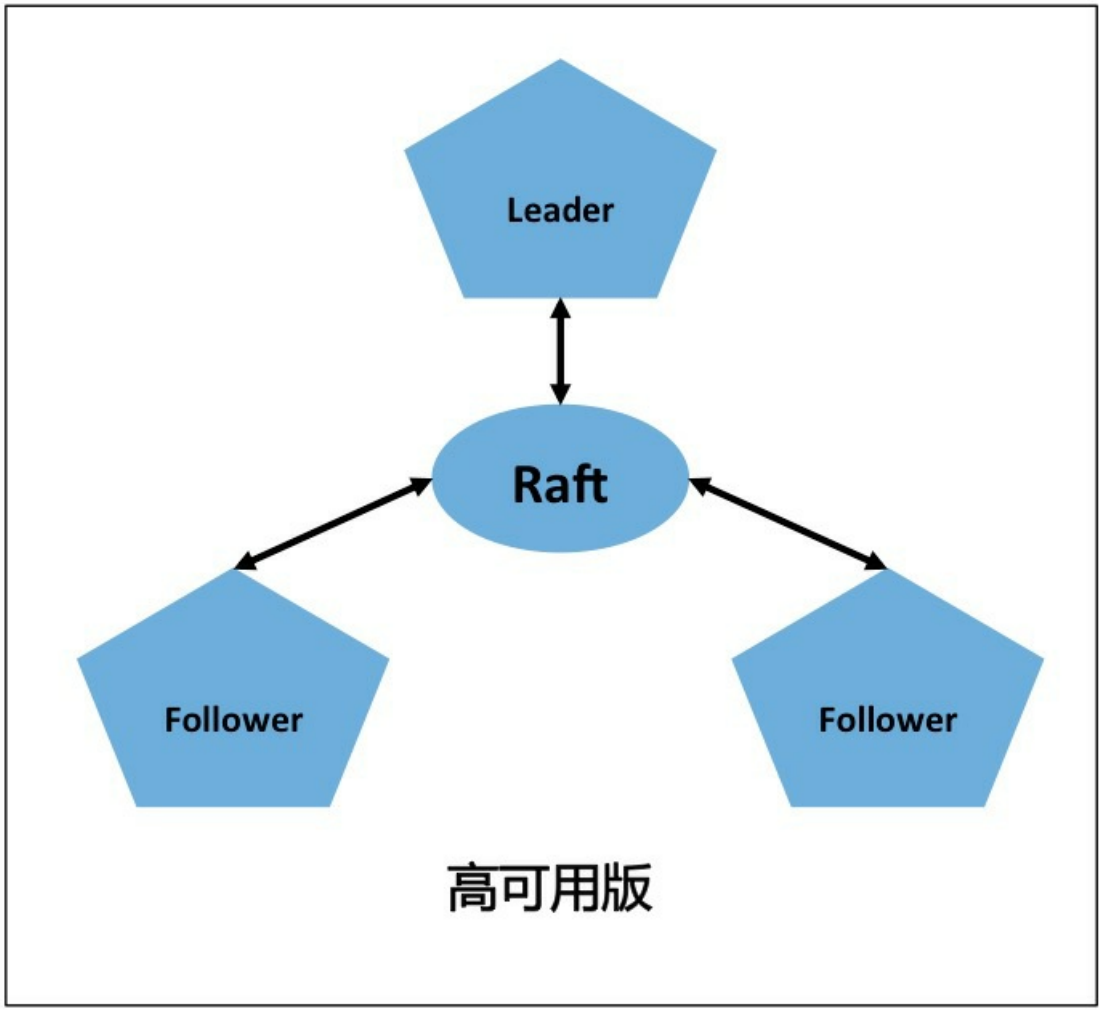
  是的，云时序数据库均采用HTTPS加密传输。

6.3. 高可用版

云时序数据库InfluxDB®的产品系列包括基础版和高可用版，本文介绍高可用版的相关信息。

高可用版是适用性较广的云数据库系列。采用Raft三节点的高可用架构，适合80%以上的用户场景，包括互联网基础资源监控，容器监控，业务运营监控分析，物联网设备远程实时监控，工业安全生产监控，生产质量评估和故障回溯。

拓扑图



优势

高可用性

- 利用分布式一致性协议（Raft）保障多节点状态切换的可靠性。
- 采用三副本方案，数据日志从主节点同步复制到两个备节点，当集群中至少两个节点都写入成功后，数据写入才完成提交。
- 节点间采用线性一致性写。

限制

基于性能考虑，高可用版实例的三个节点均在同一可用区。

常见问题

1. 基础版可以升级为高可用版本吗？暂不支持。
2. 高可用版本可以降为基础版吗？暂不支持。您可以购买基础版本实例后将数据迁移，然后释放原实例。

7.快速入门

influx的命令行界面（CLI），它是一个用来跟数据库进行交互的轻量级工具。有关如何下载使用CLI，可参考文档命令行界面。CLI默认通过端口3242向TSDB For InfluxDB® HTTP API发送请求，直接与TSDB For InfluxDB®通信。

> **说明**
>
> 也可以向数据库发送原生的HTTP请求，请参阅文档通过HTTP API写入数据和通过HTTP API查询数据中使用curl的示例。

初探数据的读写功能

当在阿里云平台成功创建好数据库之后，TSDB For InfluxDB®就可以接收写入和查询请求。

首先，对数据库存储做个简短的入门介绍。TSDB For InfluxDB®里的数据按“时间序列”来组织，其包含一个被测量的指标，如“cpu_load”或“temperature”。时间序列有零个或多个数据点，每个数据点都是一个测量值。数据点由 `time` （一个时间戳）、 `measurement` （测量指标，例如“cpu_load”）、至少一个key-value格式的 `field` （测量值，例如“value=0.64”或者“temperature=21.2”）和零或多个包含测量值元数据的key-value格式的 `tag` （例如“host=server01”，“region=EMEA”，“dc=Frankfurt”）组成。

从概念上来看，您可以将 `measurement` 看成是一个SQL表格，其中，时间戳始终是主索引， `tag` 和 `field` 是表格中的列， `tag` 会被建索引，而 `field` 则不会。与SQL表格的不同之处在于，使用TSDB For InfluxDB®，您可以有数百万的measurement，无需预先定义数据的schema，并且不会存储空值。

数据写入TSDB For InfluxDB®使用行协议（Line Protocol），该协议遵循以下格式：

```
<measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value>[,<field2-key>=<field2-value>...] [unix-nano-timestamp]
```

以下是符合格式的数据写入TSDB For InfluxDB®的示例：

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i 1434067467100293230
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 1434067467000000000
```

> **说明**
>
> 更多关于行协议的信息可参考文档行协议参考。

使用CLI写入一个数据点到TSDB For InfluxDB®，请先输入 `INSERT` ，然后输入该数据点的信息：

```
> INSERT cpu,host=serverA,region=us_west value=0.64
>
```

现在，一个measurement为 `cpu` ，tag为 `host` 和 `region` ，测量值 `value` 为 `0.64` 的数据点已经写入数据库。

查询刚刚写入的数据：

```
> SELECT “host”, “region”, “value” FROM “cpu”
> name: cpu
------------
time                            host       region    value
2015-10-21T19:28:07.580664347Z  serverA    us_west   0.64

>
```

> **说明**
>
> 前面我们在写入数据点的时候没有提供时间戳。如果写入没有带时间戳的数据点，TSDB For InfluxDB® 会在获取该点时，把本地当前时间分配给该数据点，作为该数据点的时间戳。这意味着您的时间戳跟上面的会有所不同。

让我们尝试写入另一种类型的数据，同一个measurement有两个field：

```
> INSERT temperature,machine=unit42,type=assembly external=25,internal=37
>
```

查询的时候，若想返回所有的field和tag，可以使用操作符 `*` ：

```
> SELECT * FROM "temperature"
name: temperature
-----------------
time                            external  internal  machine  type
2015-10-21T19:28:08.385013942Z  25        37        unit42   assembly

>
```

> **注意**
>
> 在大型数据库上使用*而不使用LIMIT子句可能会导致性能问题。 您可以使用Ctrl+C取消响应时间过长的查询。

TSDB For InfluxDB®有很多功能和特性没有在这里提及，包括支持Go语言风格的正则表达式，例如：

```
> SELECT * FROM /.*/ LIMIT 1
--
> SELECT * FROM “cpu_load_short”
--
> SELECT * FROM “cpu_load_short” WHERE “value” > 0.9
```

这就是你在入门指南里需要知道的将数据写入TSDB For InfluxDB®并进行查询的全部内容。想要获取更多关于TSDB For InfluxDB®写入数据和查询数据的信息，请查看文档通过HTTP API写入数据和通过HTTP API查询数据。想要了解更多TSDB For InfluxDB®中涉及的概念，请查看文档关键概念。

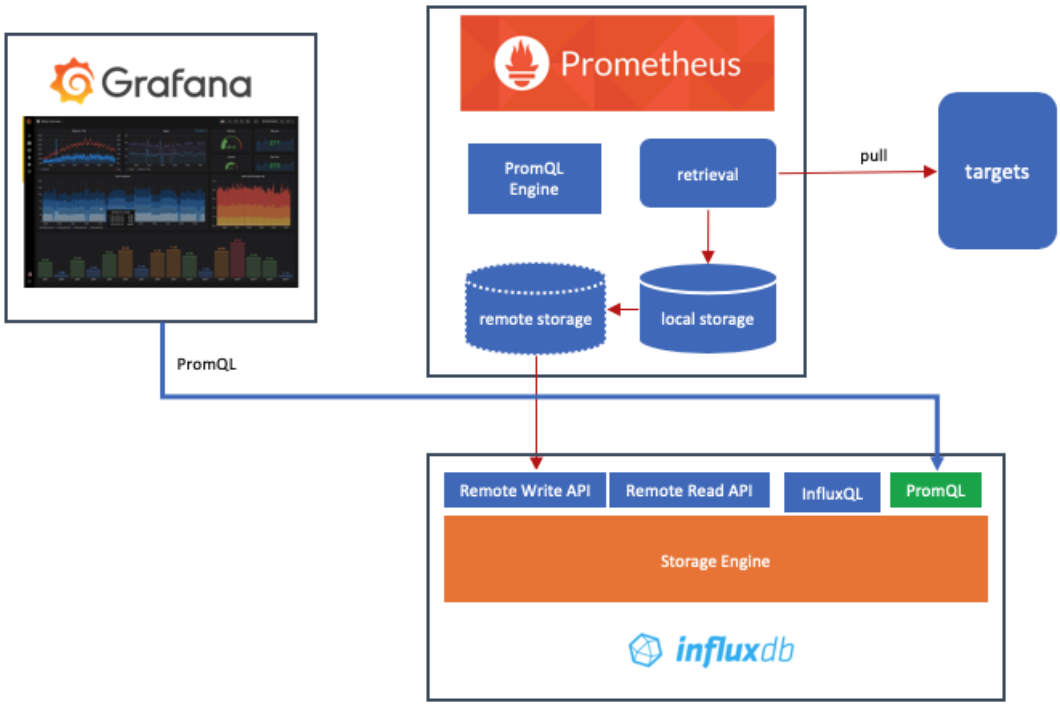
8.Prometheus生态

8.1. 配置Grafana数据源

为了更好的服务Prometheus生态，阿里云InfluxDB®集成了标准的Prometheus查询功能，用户可以直接使用PromQL查询InfluxDB®中的数据。本文介绍通过配置Grafana数据源体验PromQL查询功能。

背景信息

Prometheus已成为云原生监控领域的事实标准，而InfluxDB®提供的存储能力非常适合作为Prometheus的长期存储方案。对于数据查询，Prometheus的Remote Read API可以从InfluxDB®读取数据，但是该模式链路较长，当数据量较大时对InfluxDB®和Prometheus都有内存压力。对于Grafana用户，可以通过修改数据源的URL切换到阿里云InfluxDB®。技术架构如下图：



用户可以通过控制台升级到最新版本，以确保使用最新功能。

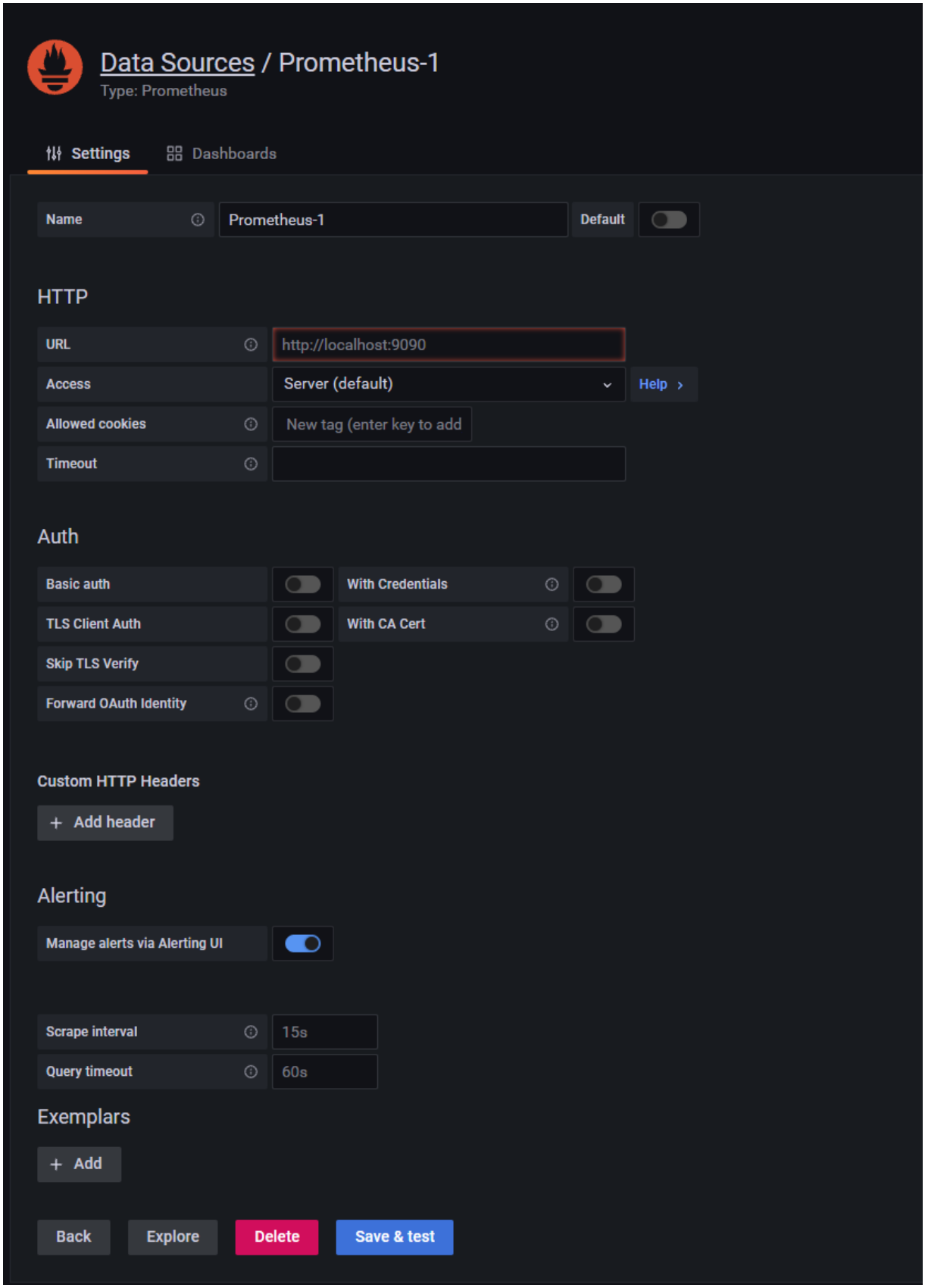
前提条件

已安装并连接Grafana服务，操作详情请参见Grafana如何接入阿里云时序数据库TSDB For InfluxDB®。

配置Grafana数据源

用户可以使用Grafana创建Prometheus数据源，也可以更新已有的数据源，不需要更新仪表盘即可体验InfluxDB提供的PromQL查询功能。

1. 在Grafana首页，单击**Add your first data source**。
2. 在添加数据源界面，选择Prometheus。
3. 在**Settings**页签，配置以下参数，单击**Save & test**。


Data Sources / Prometheus-1
Type: Prometheus
Settings
Dashboards
Name
Prometheus-1
Default
HTTP
URL
http://localhost:9090
Access
Server (default)
Help
Allowed cookies
New tag (enter key to add
Timeout
Auth
Basic auth
With Credentials
TLS Client Auth
With CA Cert
Skip TLS Verify
Forward OAuth Identity
Custom HTTP Headers
Add header
Alerting
Manage alerts via Alerting UI
Scrape interval
15s
Query timeout
60s
Exemplars
Add
Back
Explore
Delete
Save & test

参数	说明
URL	填写InfluxDB实例的VPC网络地址或公共网络地址。
Auth	选择使用Basic auth认证。
Basic Auth Details	填写InfluxDB的用户名和密码。
Custom query parameters	填写指定查询的InfluxDB数据库名称。

9.Grafana监控可视化

Grafana是一款流行的开源时序数据可视化工具，在互联网应用分析、工业监控、气象监控、家居自动化和过程管理等领域都有着广泛的应用。阿里云InfluxDB®服务集成了Grafana，用户无需安装和部署就可以使用Grafana服务，并且可以基于InfluxDB中存储的时序数据快速创建仪表盘（Dashboard）。本文将介绍如何使用阿里云InfluxDB®服务提供的Grafana可视化工具。

前提条件

已购买InfluxDB®实例，并创建数据库和访问账号，操作详情请参见购买流程和管理用户账号和数据库。

步骤一：开启Grafana公网访问链接

1. 登录阿里云官网。
2. 在页面左上角，选择目标地域。
3. 在目标实例操作列单击**管理**。
4. 在左侧导航栏，选择**可视化Grafana > 服务访问和账号**。
5. 在**服务访问和账号**页面，单击**开通Grafana公网访问链接**，开启Grafana公网访问链接。

步骤二：创建Grafana访问账号

1. 在**服务访问和账号**页面，单击**创建账号**。

2. 在弹出的对话框，输入Grafana用户名和密码，单击**创建**，完成账号创建。

步骤三：创建Grafana数据源

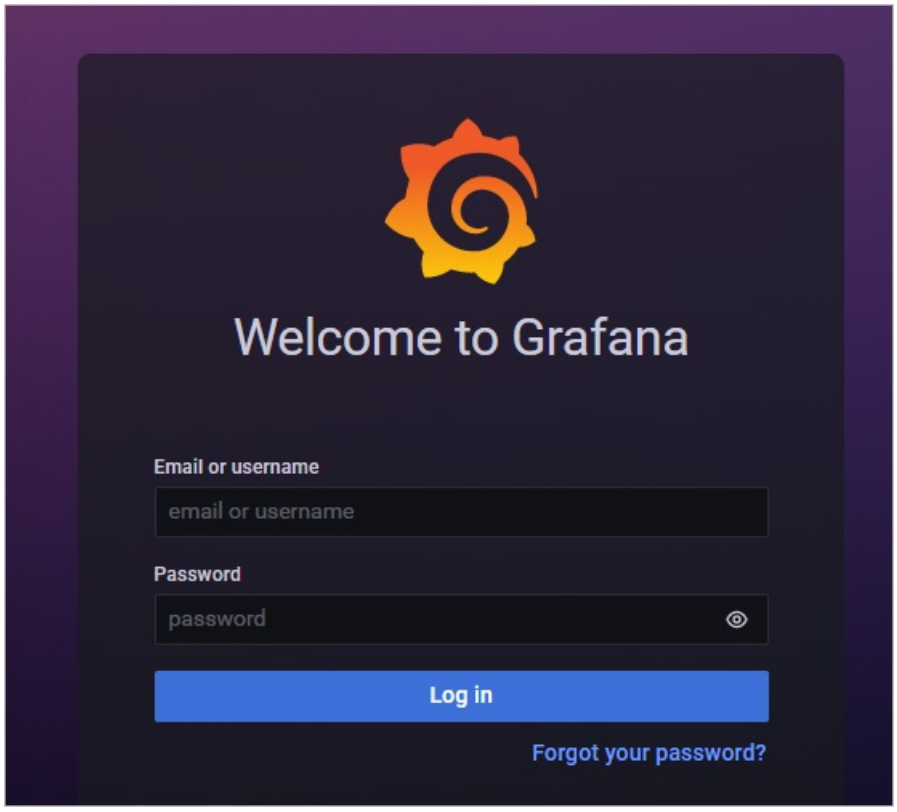
1. 在左侧导航栏，选择**可视化Grafana > 数据源管理**。
2. 在**数据源管理**页面，单击**添加数据源**。

3. 在弹出的对话框，输入数据源的名称，选择InfluxDB®数据库和访问账号，输入InfluxDB的账号密码，然后点击**测试连接并保存**即可完成创建。

步骤四：登录使用Grafana

1. 在左侧导航栏，选择**可视化Grafana > 服务访问和账号**。
2. 在**服务访问和账号**页面，单击**登录Grafana**。

3. 在**Grafana登录**页面，输入Grafana用户账号和密码登录。


Welcome to Grafana
Email or username
email or username
Password
password
Log in
Forgot your password?

10.最佳实践

10.1. TSDB for InfluxDB®☺与自建InfluxDB对比优势

InfluxDB® is a trademark registered by InfluxData, which is not affiliated with, and does not endorse, TSDB for InfluxDB®.

特性对比

对比项	自购硬件搭建InfluxDB	自购ECS搭建InfluxDB	TSDB for InfluxDB®
数据可靠性	可靠性较低，即使自建RAID仍然有不少风险。	基于高效云盘的99.9999999%可靠性。	基于高效云盘的99.9999999%可靠性。
服务可用性	可用性较低，会因为电力、服务器、磁盘甚至程序本身等各种因素导致服务中断。	容易因为参数设置不当或瞬时压力造成进程终止，且需要自己搭建完善的进程恢复机制。另外，很难处理ECS故障导致的服务终止。	根据规格和场景提供最优的参数配置，提供完善的进程恢复机制以及全自动的ECS故障转移机制。另外，具备比开源版本InfluxDB更完善的内存管理机制，实现了全自动Load Shedding，从容应对瞬时冲击。具有磁盘即将写满的自动保护机制。
系统安全性	自行部署，价格高昂；自行修复数据库安全漏洞。	自行部署，价格高昂；自行修复数据库安全漏洞。	防DDoS攻击，流量清洗；及时修复各种数据库安全漏洞。
运维难度	软硬件都需维护，运维困难非常多。	需要专职DBA维护，且熟悉InfluxDB的DBA较少，不易找到。	提供了图形化的控制台，方便进行基本的管理操作，避免学习InfluxDB复杂的基本管理命令。且提供7x24小时专业运维与答疑，彻底摆脱运维烦恼。
数据备份	自行实现，但需要寻找备份存放空间以及定期验证备份是否可恢复。	自行实现，但需要寻找备份存放空间以及定期验证备份是否可恢复。	自动实现（待上线）。
冷热数据分离	现有开源InfluxDB架构几乎无法实现。	现有开源InfluxDB架构几乎无法实现。	自动实现（待上线）。

价格对比

对比项	自购硬件搭建InfluxDB	自购ECS搭建InfluxDB	TSDB for InfluxDB®
初始一次性投入费用（4C8G硬件规格为例）	低配服务器至少约7000元起，还要加上交换机、机架甚至机房等硬件设施投入的费用。	3年自购4C8G规格ECS的费用约5000元左右，且没有任何其它的硬件和基础设施费用。而且也可以选择按月付费，避免短期资金压力。	一般情况下一次性投入会比自购ECS更低。而且也可以选择按月付费，避免短期资金压力。
持续性费用	电力费用、机房费用、带宽费用、硬件损耗费用等都是不小的数目。而且还需要不菲的DBA人工费用。	需要不菲的DBA人工费用。	无任何持续性费用。

10.2. 使用Influx CLI（命令行界面）连接TSDB For InfluxDB®

通过Influx CLI，用户可以方便地连接到TSDB For InfluxDB®，进行数据读写。本文主要介绍如何使用Influx CLI连接到阿里云上的TSDB For InfluxDB®，以及如何通过CLI写入数据和查询数据。

准备工作

首先，需要在阿里云官网上购买TSDB For InfluxDB®实例，并成功创建用户账号。Influx CLI连接TSDB For InfluxDB®需要用户账号和密码。

下载CLI

1. 进入到下载页面，选择InfluxDB v1.7.6。
2. 根据操作系统的不同，下载合适的二进制包，如下图所示。
3. 解压下载好的二进制包。Mac OS X以及Linux操作系统，在目录usr/bin/下可获得命令行工具influx。对于Windows操作系统，进入到解压后的文件夹即可获得二进制文件influx.exe。例如，如果用户使用的是Mac OS X操作系统，可按如下方式下载和进入influx所在的目录：

```
wget https://dl.influxdata.com/influxdb/releases/influxdb-1.7.6_darwin_amd64.tar.gz
tar zxvf influxdb-1.7.6_darwin_amd64.tar.gz
cd influxdb-1.7.6-1/usr/bin
```

通过CLI连接TSDB For InfluxDB®

打开终端，输入如下命令即可连接到您所购买的实例：

```
./influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242
```

TSDB For InfluxDB®使用SSL协议保障数据传输过程中的安全性。“-ssl”，“-username”，“-password”，“-host”和“-port”这五个参数是必需的，在每次连接TSDB For InfluxDB®的时候都需要提供。

参数解释

- `-ssl`：指的是使用HTTPS连接InfluxDB服务器。
- `-username`：指的是已创建好的用户账号。
- `-password`：指的是用户账号对应的密码。
- `-host`：指的是网络地址，在管理控制台的实例详情中可查看，如下图所示。
- `-port`：指的是网络端口，默认是3242。

通过CLI写入数据

在写入数据前，需要先创建数据库，详情请参见创建数据库，假设已成功创建名为mydb的数据库。

- **单点写入**
  i. 通过Influx CLI连接TSDB For InfluxDB®。

  ```
  ./influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242
  ```

  ii. 指定使用数据库mydb。

  ```
  > USE mydb
  Using database mydb
  ```

  iii. 将单个数据点写入数据库mydb。

  ```
  > INSERT cpu,host=serverA,region=us_west value=0.64
  ```

  这条语句将一个数据点写入数据库mydb的默认保留策略中。其中，`cpu`是measurement，`host`和`region`是tag，`value`是field，数据点的格式符合行协议。如果没有提供时间戳，系统会把服务器本地的纳秒级的Unix时间当作数据点的时间戳。在TSDB For InfluxDB®中，任何时间戳都是UTC时间（协调世界时）。

- **文件导入**
  i. 示例数据：文件test.txt，里面包含多个数据点。

```
# DML
# CONTEXT-DATABASE: mydb
# CONTEXT-RETENTION-POLICY: autogen
h2o_feet,location=coyote_creek water_level=3.524,level\ description="between 3 and 6
feet"1439868600
h2o_feet,location=coyote_creek water_level=3.399,level\ description="between 3 and 6
feet"1439868960
h2o_feet,location=coyote_creek water_level=3.278,level\ description="between 3 and 6
feet"1439869320
h2o_feet,location=coyote_creek water_level=3.159,level\ description="between 3 and 6
feet"1439869680
h2o_feet,location=coyote_creek water_level=3.048,level\ description="between 3 and 6
feet"1439870040
h2o_feet,location=coyote_creek water_level=2.943,level\ description="below 3 feet"143
9870400
h2o_feet,location=coyote_creek water_level=2.831,level\ description="below 3 feet"143
9870760
h2o_feet,location=coyote_creek water_level=2.717,level\ description="below 3 feet"143
9871120
h2o_feet,location=coyote_creek water_level=2.625,level\ description="below 3 feet"143
9871480
h2o_feet,location=coyote_creek water_level=2.533,level\ description="below 3 feet"143
9871840
```

ii. 将文件中的数据导入TSDB For InfluxDB®。

```
./influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242-import-path
=test.txt
```

使用参数 `-import` 可将文件中的数据导入TSDB For InfluxDB®。其中， `-path` 指定需要被导入的文件。这条语句将 `measurement` 为 `h2o_feet` ， `tag` 为 `location` ， `field` 为 `water_level` 和 `level description` 的数据写入数据库mydb中。

通过CLI查询数据

TSDB For InfluxDB®支持丰富的InfluxQL查询语句，InfluxQL是一种类似SQL的语言，能够让用户方便快捷地与数据进行交互，详情可查看文档数据探索和Schema探索。

1. 在进行数据查询前，先连接TSDB For InfluxDB®。

```
./influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242
```

2. 指定从哪个数据库中查询数据，在本文的示例中，数据来自数据库“mydb”。

```
> USE mydb
Using database mydb
```

查询指定measurement中的所有数据

```
> SELECT * FROM "h2o_feet"
name: h2o_feet
time       level description    location     water_level
----------------------------------------
1439868600 between 3and6 feet coyote_creek 3.524
1439868960 between 3and6 feet coyote_creek 3.399
1439869320 between 3and6 feet coyote_creek 3.278
1439869680 between 3and6 feet coyote_creek 3.159
1439870040 between 3and6 feet coyote_creek 3.048
1439870400 below 3 feet         coyote_creek 2.943
1439870760 below 3 feet         coyote_creek 2.831
1439871120 below 3 feet         coyote_creek 2.717
1439871480 below 3 feet         coyote_creek 2.625
1439871840 below 3 feet         coyote_creek 2.533
```

该查询返回名为“h2o_feet”的measurement对应的所有数据。

查询measurement中特定的field并进行算术运算

```
> SELECT ("water_level"+2)*3 FROM "h2o_feet"
name: h2o_feet
time       water_level
---------------
143986860016.572
143986896016.197
143986932015.834000000000001
143986968015.477
143987004015.144
143987040014.828999999999999
143987076014.492999999999999
143987112014.151000000000002
143987148013.875
143987184013.598999999999998
```

“water_level”是“h2o_feet”中的一个field，该查询将“water_level”中的每个值加2，然后乘以3。请注意，TSDB For InfluxDB®遵循标准的算术运算顺序，可查看数学运算符了解更多相关信息。

查询field value满足一定条件的数据

```
> SELECT * FROM "h2o_feet" WHERE "water_level">3
name: h2o_feet
time       level description    location     water_level
----------------------------------------
1439868600 between 3and6 feet coyote_creek 3.524
1439868960 between 3and6 feet coyote_creek 3.399
1439869320 between 3and6 feet coyote_creek 3.278
1439869680 between 3and6 feet coyote_creek 3.159
1439870040 between 3and6 feet coyote_creek 3.048
```

该查询返回中“h2o_feet”中的数据，这些数据满足条件：“water_level”的值大于3。更多关于WHERE子句的介绍可查看WHERE语句。

将查询结果分组

```
> SELECT MEAN("water_level") FROM "h2o_feet" GROUP BY "location"

name: h2o_feet
tags: location=coyote_creek
time mean
--------
03.0057
```

该查询将查询结果按“location”分组，并计算每个“location”中的“water_level”的平均值。更多关于GROUP BY子句的介绍可查看GROUP BY子句。

使用正则表达式指定field key和tag key

```
> SELECT /level/ FROM "h2o_feet"
name: h2o_feet
time       level description    water_level
--------------------------------
1439868600 between 3and6 feet 3.524
1439868960 between 3and6 feet 3.399
1439869320 between 3and6 feet 3.278
1439869680 between 3and6 feet 3.159
1439870040 between 3and6 feet 3.048
1439870400 below 3 feet         2.943
1439870760 below 3 feet         2.831
1439871120 below 3 feet         2.717
1439871480 below 3 feet         2.625
1439871840 below 3 feet         2.533
```

该查询返回所有包含字符串“level”的field key和tag key对应的数值。更多关于正则表达式的介绍可查看正则表达式。

10.3. 从采集到分析-TSDB For InfluxDB® 让你的数据产生价值

数据无处不在，价值无处不在。在时序数据库领域，TSDB For InfluxDB®作为一款数据存储分析的利刃，在生产和开发环境中得到了比较广泛的应用。本文主要讲述如何将你的数据搬迁上云，让你的数据产生更大的价值。

了解我们-TSDB For InfluxDB®

TSDB For InfluxDB®是一款专门处理高写入和查询负载的时序数据库，用于存储大规模的时序数据并进行实时分析，包括来自DevOps监控、应用指标和IoT传感器上的数据。它的主要特点如下：

- 专为时间序列数据量身订造高性能数据存储。TSM引擎提供数据高速读写和压缩等功能。
- 简单高效的HTTP API写入和查询接口。
- 针对时序数据，量身订造类似SQL的查询语言，轻松查询聚合数据。
- 允许对tag建索引，实现快速有效的查询。
- 数据保留策略（Retention policies）能够有效地使旧数据自动失效。

在阿里云购买成功之后，我们需要创建数据库账户和密码，设置存储策略。通过实例详情页面可以查看访问TSDB For InfluxDB®的方式，目前提供两种HTTP访问地址：VPC网络（什么是专有网络，相同VPC内的云产品实例可以相互访问）与公共网络（需购买时开通，内外部网络都可访问），均开通了HTTPS加密验证，**注意接入时使用HTTPS通道**。

实例生产完成后可以在管理用户账号和数据库用户test和数据库test，并保证用户test对数据库test具有读写权限。

Telegraf-数据的搬运工

Telegraf是一个用Go语言编写的代理程序，可采集系统和服务的统计数据，并写入TSDB For InfluxDB®数据库。Telegraf具有内存占用小的特点，通过插件系统开发人员可轻松添加支持其他服务的扩展。Telegraf安装完成后，数据可以用HTTP方式写入TSDB For InfluxDB®。

urls选择实例详情中的公网或者VPC访问地址，修改上图标记中的写入的数据库和存储策略，存储策略在不填的情况下将会写入该数据库默认存储策略中。TSDB For InfluxDB®采用认证的方式写入，在填写username和password之前需要确保该账号对数据库具有写权限。关于权限可以在实例管理->账号管理中进行设置。完成这些操作之后，Telegraf采集的数据便可以写入到TSDB For InfluxDB®中。

启动Telegraf

Ubuntu, Debian, RedHat, CentOS启动

```
sudo service telegraf start
```

Ubuntu 15.04+, Debian 8+, CentOS 7+, RHEL 7+启动

```
sudo systemctl start telegraf
```

数据迁移-传输的高速通道

对于在自建环境或者其它平台中已经拥有InfluxDB实例的用户，如果考虑将业务迁移至TSDB For InfluxDB®服务中，我们打造了一键迁移的工具，具体实践可以参考数据迁移。注意迁移工具会自动在TSDB For InfluxDB®上创建选择迁移的数据库，迁移之前不需手动创建，否则会存在写冲突。数据迁移之后，数据库运维工作放心交给系统后台，用户可以节省更多的时间写入新数据和对实例中的数据进行查询分析。

客户端写入-不一样的风格

InfluxDB开源社区提供了丰富的SDK，基本上涵盖了主流了编程语言，TSDB For InfluxDB®完全兼容各种客户端。下面将介绍几种主要的写入方式：

Go写入方式

```
package main

import(
"fmt"
"log"
"math/rand"
"net/url"
"time"

    client "github.com/influxdata/influxdb1-client"
)
```

```
func main(){
    host, err := url.Parse(fmt.Sprintf("https://%s:%d","xxx.influxdata.rds.aliyuncs.com",32
42))
if err !=nil{
        log.Fatal(err)
}
    config := client.Config{
        URL:*host,
Username:"test",
Password:"test",
}
    con, err := client.NewClient(config)
if err !=nil{
        log.Fatal(err)
}

    _, _, err = con.Ping()
if err !=nil{
        log.Fatal(err)
}

var(
        shapes     =[]string{"circle","rectangle","square","triangle"}
        colors     =[]string{"red","blue","green"}
        sampleSize =1000
        pts        = make([]client.Point, sampleSize)
)

    rand.Seed(42)
for i :=0; i < sampleSize; i++{
        pts[i]= client.Point{
Measurement:"shapes",
Tags: map[string]string{
"color": colors[rand.Intn(len(colors))],
"shape": shapes[rand.Intn(len(shapes))],
},
Fields: map[string]interface{}{
"value": rand.Intn(sampleSize),
},
Time:      time.Now(),
}
}

    bps := client.BatchPoints{
Points:          pts,
Database:"test",
RetentionPolicy:"autogen",
}
    _, err = con.Write(bps)
if err !=nil{
        log.Fatal(err)
}
}
```

HTTP连接配置中需要指明实例的公网或者VPC地址和数据库用户、密码（具体可以在实例管理中查询）。在写入前指定数据写入的数据库和保留策略（如果不指定，则写入默认的保留策略），数据可以按行或者Batch的方式写入。为了提高写入吞吐，推荐Batch的写入方式，Batch行数按数据大小可以在100~1000行左右。SDK下载，请参见地址，程序编译执行命令为 `go run main.go` 。

JAVA写入方式

```
package main;

import java.util.concurrent.TimeUnit;
import org.influxdb.InfluxDB;
import org.influxdb.BatchOptions;
import org.influxdb.InfluxDBFactory;
import org.influxdb.dto.Point;


publicclassStartMain{

publicstaticvoid main(String[] args)throwsException{
StartMain startMain =newStartMain();
try{
            startMain.run();
}catch(Exception e){
System.out.println(e);
}
}

publicvoid run()throwsException{

InfluxDB influxDB =InfluxDBFactory.connect("https://xxx.influxdata.rds.aliyuncs.com:3242","
test","test");

String dbName ="test";
        influxDB.setDatabase(dbName);
String rpName ="autogen";
        influxDB.setRetentionPolicy(rpName);
        influxDB.enableBatch(BatchOptions.DEFAULTS);

        influxDB.write(Point.measurement("cpu")
.time(System.currentTimeMillis(),TimeUnit.MILLISECONDS)
.addField("idle",90L)
.addField("user",9L)
.addField("system",1L)
.build());

        influxDB.write(Point.measurement("disk")
.time(System.currentTimeMillis(),TimeUnit.MILLISECONDS)
.addField("used",80L)
.addField("free",1L)
.build());

        influxDB.close();
}
}
```

Java SDK下载见地址，在 `enableBatch` 中设置异常捕捉函数可以处理写入异常。

```
influxDB.enableBatch(BatchOptions.DEFAULTS.exceptionHandler(
(failedPoints, throwable)->{/* custom error handling here */})
);
```

Influx CLI写入

TSDB for InfluxDB®支持命令行界面Influx CLI，连接时需指定ssl、host、port、username和password：

```
influx -ssl -host xxx.influxdata.rds.aliyuncs.com -port 3242-username admin -password admin
```

客户端上通过行写入协议可以将数据写入TSDB For InfluxDB®，如：

```
> INSERT cpu,host=serverA,region=us_west value=0.64
```

Shell写入

```
curl -i -XPOST -u test:test  'https://xxx.influxdata.rds.aliyuncs.com:3242/write?db=test'--
data-binary 'cpu_load_short,host=server01,region=us-west value=0.64 1434055562000000000'
```

Shell的具体用法可以参考HTTP API。

Shell的具体用法可以参考HTTP API。

数据分析

目前TSDB For InfluxDB®支持丰富的类SQL查询，非常方便从业者使用。

下面示例是查询每组 `color` 的平均值：

```
> SELECT MEAN("value") FROM "shapes" GROUP BY "color"
name: shapes
tags: color=blue
time mean
--------
0216.25

name: shapes
tags: color=green
time mean
--------
0434.25

name: shapes
tags: color=red
time mean
--------
0540.25
```

10.4. Grafana接入阿里云时序数据库TSDB For InfluxDB®服务

Grafana是在互联网架构和应用分析中最流行的时序数据可视化工具，并且也在工业监控、气象监控、家居自动化和过程管理等领域有着广泛的应用。将阿里云时序数据库TSDB For InfluxDB®接入Grafana后，您可以利用Grafana的丰富易用的可视化工具更好地监控和分析来自阿里云时序数据库TSDB For InfluxDB®的数据。本文介绍如何将阿里云时序数据库TSDB For InfluxDB®接入Grafana。

前置条件

在接入Grafana之前，若您已经购买阿里云时序数据库TSDB For InfluxDB®并配置好用户账号与数据库信息，同时写入了一定量的数据。如果您还没有购买，可以参考购买流程。购买完成后，您还需要创建用户账号、数据库和存储策略，并授予账号访问数据库的权限，这里可以参考管理用户账号和数据库。最后您还需要向数据库写入一定的数据，以便通过Grafana生成可视化的监控图表。具体的写入方式可以参考通过HTTP API写入数据，行协议参考，和行协议教程。

操作步骤

1. 下载与安装Grafana。

   访问Grafana官方网站的下载地址，这个地址中详细介绍了各种操作系统下Grafana的安装步骤。根据自己的系统版本和配置，下载对应的安装包进行安装。

   以Centos为例，具体的下载与安装命令如下：

   ```
   wget https://dl.grafana.com/oss/release/grafana-6.1.4-1.x86_64.rpm
   sudo yum localinstall grafana-6.1.4-1.x86_64.rpm
   ```

2. 启动与登录Grafana服务。

   下载安装完成后，输入对应操作系统的启动命令来启动Grafana服务。

   以Centos为例：

   ```
   service grafana-server start
   ```

   以Mac为例：

   ```
   brew services start grafana
   ```

   启动Grafana服务后，打开浏览器，输入IP和端口，3000为Grafana的默认侦听端口。如果您是在本地搭建的Grafana服务，可以访问127.0.0.1:3000；如果您是在阿里云ECS上搭建的Grafana服务，请在阿里云ECS控制台上查阅您的ECS公网访问地址。

   Grafana的默认管理员账号密码为admin/admin，首次登录可根据您的需要修改管理员默认密码。

3. 配置数据源

   主页单击**Add data source**，进入新的数据源设置界面。

4. 在数据源设置界面，配置以下参数。

参数	说明
Type	选择 InfluxDB。
HTTP URL	您实例的公共网络地址。 说明 要查看该地址，请在阿里云时序数据库TSDB For InfluxDB®控制台中，选择**实例列表 > 管理**，查看公共网络地址一栏。
InfluxDB Details	要访问的数据库和对应数据库的用户账号、密码等信息。

5. 配置Dashboard。

新建Dashboard

Dashboard是Grafana可视化展示的重要组件。根据配置的查询规则，Grafana会向阿里云时序数据库TSDB For InfluxDB®获取数据，并展示在Dashboard上。本文以Graph类型为例，配置一个Dashboard。

i. 单击添加Dashboard。

ii. 选择Graph类型。

编写查询规则

i. 单击Graph的标题栏，选择编辑查询规则。

ii. 进入详细的查询规则配置页面。 我们可以看到这里的选项很多，具体填写步骤参考如下：

- Data Source处选择配置好的Data Source。
- 在FROM后方先写的两个值分别为存储策略与MEASUREMENT。
- WHERE后填写的是tag filter，可添加多个。
- SELECT后是field与聚合函数的查询条件，特别是如果配置了一个time的GROUP BY条件的话，您需要在这里配置聚合函数。聚合函数的执行顺序是从左往右的，比如：

  这样配置后，在阿里云时序数据库TSDB For InfluxDB®内的select字句是这样的：

  ```
  SELECT derivative(mean("field10"),10s)/10 AS "REQ/s" FROM ....
  ```

- GROUP BY可以配置group by时间或group by tag，也可以配置排序与limit。
- FORMAT AS配置的是显示方式。
- ALIAS BY后配置的是measurement或tag的别名。
- Add Query可以配置多个子查询。

iii. 配置完成后，单击页面上方的保存按钮。

展示效果

根据您配置的查询规则，数据会即时显示在Graph中，根据查询规则的不同，展示效果也有所区别。效果可以参考以下示例图。

10.5. Prometheus对接阿里云TSDB For InfluxDB®服务

Prometheus是指标监控领域的领军软件，其提供的远程存储接口可将InfluxDB作为时序数据的存储，解锁本地存储的限制，提供更稳定可靠的用户体验。本文将介绍如何使用阿里云的TSDB For InfluxDB®服务来对接Prometheus，构建监控数据的高效存储方案。

背景信息

Prometheus是一套监控和告警系统，最初由SoundCloud公司开发，2016年加入CNCF，成为流行的开源项目。Prometheus支持多维数据模型，使用metric和label来表示时序数据，同时提供了PromQL查询语言来支持多维查询。存储方面，Prometheus内置一个基于本地存储的时序数据库，这个单节点的数据库在扩展性和可靠性方面都有所限制，功能也不够强大；Prometheus没有在存储系统深入造轮子，而是通过提供接口的方式来集成其他存储系统，建立了灵活多样的生态。Prometheus的远端存储（remote storage）原理如下图所示：

Prometheus定义了同远端存储的读写接口，交互协议使用protocol buffer定义，传输基于HTTP；一个存储系统如果要支持Prometheus，仅需要实现一个adapter层，将Prometheus的读写请求转换为其内部的格式来处理。TSDB For InfluxDB®时序数据库提供高性能数据存储功能，支持数据保留策略等特性，是Prometheus远端存储很好的选择TSDB For InfluxDB®内置了对Prometheus协议的支持，集成了adapter的功能，直接提供了两个HTTP API来处理Prometheus读写请求。关于TSDB For InfluxDB®时序数据库详细说明，请参见InfluxDB®介绍。

- /api/v1/prom/read
- /api/v1/prom/write

前提条件

- 已购买TSDB For InfluxDB®实例，请参见购买流程。
- 已创建数据库与用户，请参见管理用户账号和数据库。
- 已安装Prometheus，请参见官方文档。

配置 Prometheus

> **说明**
>
> Prometheus支持多个平台，本文以Linux系统为例来介绍

Linux平台的Prometheus下载解压后，包含了一个配置文件，我们仅需要在配置文件中增加InfluxDB远端存储的配置。

```
$ ls
console_libraries  consoles  data  LICENSE  NOTICE  prometheus  prometheus.yml  promtool
```

使用文本编辑器（比如vim）修改**prometheus.yml**, 在文件末尾增加下面几行：

> **说明**
>
> 其中URL地址替换为真实的TSDB For InfluxDB®实例的公网地址，u值请替换为已创建的用户，p值请替换为用户密码。

```
remote_write:
    - url: "https://ts-1234abcd.influxdata.rds.aliyuncs.com:3242/api/v1/prom/write?db=prometheus&u=prom&p=mypassword"

remote_read:
    - url: "https://ts-1234abcd.influxdata.rds.aliyuncs.com:3242/api/v1/prom/read?db=prometheus&u=prom&p=mypassword"
```

修改完成后保存该配置文件，然后启动Prometheus：

```
$./prometheus --config.file=prometheus.yml
```

启动成功后，就可以通过浏览器来访问prometheus服务，其默认端口为9090.

参考文档

- 阿里云TSDB For InfluxDB®InfluxDB®☺介绍
- Prometheus endpoints support in InfluxDB
- Prometheus远端存储

11.故障排查

本文介绍常见的TSDB For InfluxDB®错误信息和关于它们的描述，以及常见的解决方案。

如何处理报错“database name required”？

当包含 `SHOW` 的查询没有明确指定一个数据库时，错误 `database name required` 就会发生。指定数据库的方法包括：在 `SHOW` 查询中使用 `ON` 子句，在CLI中使用 `USE <database_name>` ，或者在HTTP API请求中使用参数 `db` 。

相关的 `SHOW` 查询包括 `SHOW RETENTION POLICIES` 、 `SHOW SERIES` 、 `SHOW MEASUREMENTS` 、 `SHOW TAG KEYS` 、 `SHOW TAG VALUES` 和 `SHOW FIELD KEYS` 。

如何处理报错“max series per database exceeded: < >”？

当写入导致数据库中序列的数量超过每个数据库允许的最大序列数量时，错误 `max series per database exceeded` 就会发生。每个数据库允许的最大序列数量由购买的实例规格所决定。

`< >` 内的信息展示了那些超出 `max-series-per-database` 限制的序列的measurement和tag set。

如何处理报错“found < >, expected identifier at line < >, char < >”？

- InfluxQL语法

  当TSDB For InfluxDB®在查询中找不到预期的标识符时，错误 `expected identifier` 就会发生。标识符是连续查询名字、数据库名字、field key、measurement的名字、保留策略名字、subscription的名字、tag key和用户名。这个错误通常用于提醒您仔细检查您的查询语法。

  示例：

  ```
  > SELECT * FROM WHERE "blue"= true
  ERR: error parsing query: found WHERE, expected identifier at line 1, char 15
  ```

  该查询在 `FROM` 和 `WHERE` 之间缺少measurement的名字。

- InfluxQL关键字

  在某些情况下，当查询中某个标识符是InfluxQL关键字时，错误 `expected identifier` 就会发生。如果要查询也是InfluxQL关键字的标识符，请用双引号将标识符括起来。

  在某些情况下，当查询中某个标识符是InfluxQL关键字时，错误 `expected identifier` 就会发生。如果要查询也是InfluxQL关键字的标识符，请用双引号将标识符括起来。

  示例

  ```
  > SELECT duration FROM runs
  ERR: error parsing query: found DURATION, expected identifier, string, number, bool at li
  ne 1, char 8
  ```

在查询中，field key `duration` 是一个InfluxQL关键字。为了避免错误，请用双引号将 `duration` 括起来：

```
> SELECT "duration" FROM runs
```

如何处理报错“found < >, expected string at line < >, char < >”？

当TSDB For InfluxDB®在查询中找不到预期的字符串时，错误 `expected string` 就会发生。

如何处理报错“mixing aggregate and non-aggregate queries is not supported”？

当 `SELECT` 语句同时包含聚合函数和不使用聚合函数的field key或tag key时，错误 `mixing aggregate and non-aggregate` 就会发生。

聚合函数返回一个计算结果，对于没有被聚合的field或tag，没有明显的单个值可以返回。

示例

原始数据： measurement `peg` 有两个field（ `square` and `round` ）和一个tag（ `force` ）：

```
name: peg
---------
time                   square   round   force
2016-10-07T18:50:00Z281
2016-10-07T18:50:10Z4122
2016-10-07T18:50:20Z6144
2016-10-07T18:50:30Z7153
```

查询一：

```
> SELECT mean("square"),"round" FROM "peg"
ERR: error parsing query: mixing aggregate and non-aggregate queries is not supported
```

查询一包含一个聚合函数和一个单独的field。

`mean("square")` 返回一个聚合值，这是measurement `peg` 中的四个 `square` 值的平均值，但是从field `round` 的四个非聚合的field value中，没有明显的单个值可以返回。

查询二：

```
> SELECT mean("square"),"force" FROM "peg"
ERR: error parsing query: mixing aggregate and non-aggregate queries is not supported
```

查询二包含一个聚合函数和一个单独的tag。

`mean("square")` 返回一个聚合值，这是measurement `peg` 中的四个 `square` 值的平均值，但是从tag `force` 的四个非聚合的tag value中，没有明显的单个值可以返回。

如何处理报错“time and *influxql.VarRef are not compatible”？

当在查询中用双引号把日期时间字符串括起来时，错误 `time and \*influxql.VarRef are not compatible` 就会发生。需要用单引号把日期时间字符串括起来。

示例

用双引号把日期时间字符串括起来：

```
> SELECT "water_level" FROM "h2o_feet" WHERE "location"='santa_monica' AND time >="2015-08-18T00:00:00Z" AND time <="2015-08-18T00:12:00Z"
ERR: invalid operation: time and *influxql.VarRef are not compatible
```

用单引号把日期时间字符串括起来：

```
> SELECT "water_level" FROM "h2o_feet" WHERE "location"='santa_monica' AND time >='2015-08-18T00:00:00Z' AND time <='2015-08-18T00:12:00Z'

name: h2o_feet
time                   water_level
----------------
2015-08-18T00:00:00Z2.064
2015-08-18T00:06:00Z2.116
2015-08-18T00:12:00Z2.028
```

如何处理报错“bad timestamp”？

- 时间语法

  当行协议包含不是UNIX时间戳格式的时间戳时，错误 `bad timestamp` 就会发生。

  示例

  ```
  > INSERT pineapple value=1'2015-08-18T23:00:00Z'
  ERR:{"error":"unable to parse 'pineapple value=1 '2015-08-18T23:00:00Z'': bad timestamp"}
  ```

  上面的行协议使用了RFC3339格式的时间戳。为了避免错误，成功将数据点写入TSDB For InfluxDB®，将时间戳替换成UNIX时间戳：

  ```
  > INSERT pineapple,fresh=true value=11439938800000000000
  ```

- 行协议语法

  在某些情况下，当行协议中有更通用的语法错误时，错误 `bad timestamp` 就会发生。

  示例

  写入一：

  ```
  > INSERT hens location=2 value=9
  ERR:{"error":"unable to parse 'hens location=2 value=9': bad timestamp"}
  ```

在写入一中的行协议使用空格将measurement `hen` 和tag `location=2` 分开，而不是用逗号。TSDB For InfluxDB®把field `value=9` 当成了时间戳，所以返回错误。

为了避免错误，使用逗号（而不是空格）将measurement和tag分开：

```
> INSERT hens,location=2 value=9
```

写入二：

```
> INSERT cows,name=daisy milk_prod=3 happy=3
ERR:{"error":"unable to parse 'cows,name=daisy milk_prod=3 happy=3': bad timestamp"}
```

在写入二中的行协议使用空格将field `milk_prod=3` 和field `happy=3` 分开，而不是用逗号。TSDB For InfluxDB®把field `happy=3` 当成了时间戳，所以返回错误。

为了避免错误，使用逗号（而不是空格）将两个field分开：

```
> INSERT cows,name=daisy milk_prod=3,happy=3
```

更多信息请参考行协议教程和行协议参考。

如何处理报错“time outside range”？

当行协议中的时间戳在TSDB For InfluxDB®的有效时间范围之外时，错误 `time outside range` 就会发生。

最小的有效时间戳是 `-9223372036854775806` 或 `1677-09-21T00:12:43.145224194Z` ，最大的有效时间戳是 `9223372036854775806` 或 `2262-04-11T23:47:16.854775806Z` 。

如何处理报错“engine: cache maximum memory size exceeded”？

当写入速度过快，导致服务端cache大小短时间超过预设的门限时，错误 `cache maximum memory size exceeded` 就会发生。预设的门限由购买的实例规格决定。

12.SDK参考

时序数据库中使用了InfluxDB®的原生SDK。

关于InfluxDB®的原生SDK详情，请参见 InfluxDB®官方文档。

13.FAQ

本页面讨论了经常容易混淆的内容，以及跟其它数据库系统相比TSDB For InfluxDB®以意想不到的方式运行的地方。

管理（Administration）

- 如何识别TSDB For InfluxDB®的版本？
- shard group duration和保留策略之间的关系是什么？
- 当更改保留策略后，为什么数据没有丢失？
- 为什么TSDB For InfluxDB®无法解析微秒单位？

命令行界面（CLI）

- 如何使TSDB For InfluxDB®的CLI返回用户可读的时间戳？
- 非admin用户如何使用 `USE` 指定一个数据库？
- 如何使用TSDB For InfluxDB®的CLI将数据写入一个非默认的保留策略

数据类型

- 为什么不能查询布尔类型的field value？
- TSDB For InfluxDB®如何处理多个shard之间的field的类型差异？
- TSDB For InfluxDB®可以存储的最小和最大整数是多少？
- TSDB For InfluxDB®可以存储的最小和最大时间戳是多少？
- 如何知道存储在field中的数据类型？
- 是否可以改变field的数据类型？

InfluxQL函数

- 如何执行函数中的数学运算？
- 为什么查询将epoch 0作为时间戳返回？
- 哪些InfluxQL函数支持嵌套使用？

查询数据

- 什么决定了 `GROUP BY time()` 查询返回的时间间隔？
- 为什么查询没有返回任何数据或者只返回一部分数据？
- 为什么 `GROUP BY time()` 查询不返回发生在 `now()` 之后的时间戳？
- 是否可以对时间戳执行数学运算？
- 是否可以从返回的时间戳中识别写入精度？
- 当查询数据时，什么时候应该使用单引号，什么时候应该使用双引号？
- 为什么在创建一个新的默认（ `DEFAULT` ）保留策略后会丢失数据？
- 为什么带有 `WHERE OR` 时间子句的查询返回空结果？

- 为什么 `fill(previous)` 返回空结果？
- 为什么 `INTO` 查询会丢失数据？
- 如何查询tag key和field key名字相同的数据？
- 如何跨measurement查询数据？
- 时间戳的顺序是否重要？
- 如何 `SELECT` 有tag但没有tag value的数据？

序列和序列基数

- 为什么序列基数很重要？

写入数据

- 如何写入整型的field value？
- TSDB For InfluxDB®如何处理重复数据点？
- HTTP API需要怎样的换行符？
- 当将数据写入TSDB For InfluxDB®时，应该避免哪些文字和字符？
- 当写入数据时，什么时候应该使用单引号，什么时候应该使用双引号？
- 时间戳的精度是否重要？

如何识别TSDB For InfluxDB®的版本？

有多种方法可以识别您正在使用的TSDB For InfluxDB®版本：

curl路径/ping

```
$ curl -i 'https://<网络地址>:3242/ping?u=<账号名称>&p=<密码>'
HTTP/1.1204NoContent
Content-Type: application/json
X-Influxdb-Build: OSS
X-Influxdb-Version:1.7.x
```

启动TSDB For InfluxDB®的命令行界面：

```
$ influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242

Connected to https://<网络地址>:3242 version 1.7.x
```

shard group duration和保留策略之间的关系是什么？

TSDB For InfluxDB®将数据存储在shard group。一个shard group覆盖一个特定的时间间隔；TSDB For InfluxDB®通过查看相关保留策略的 `DURATION` 来确定时间间隔。下表列出了RP的 `DURATION` 和一个shard group的时间间隔之间的默认关系：

RP持续时间（duration）	Shard group时间间隔
< 2 days	1 hour
>= 2 days and <= 6 months	1 day
> 6 months	7 days

使用 `SHOW RETENTION POLICIES` 查看保留策略的shard group duration。

当更改保留策略后，为什么数据没有丢失？

有几个原因可以解释为什么保留策略改变后数据没有马上丢失。

第一个也是最有可能的原因是，默认情况下，TSDB For InfluxDB®每30分钟检查并强制执行一次RP。您可能需要等到下一次RP检查，TSDB For InfluxDB®才能删除在RP的新 `DURATION` 之外的数据。

第二个可能的原因是，更改RP的 `DURATION` 和 `SHARD DURATION` 会导致意外的数据保留。TSDB For InfluxDB®将数据存储在shard group，每个shard group覆盖一个特定的RP和时间间隔。当TSDB For InfluxDB®强制执行RP时，整个shard group的数据会被删除，而不是单个数据点。TSDB For InfluxDB®不能拆分shard group。

如果RP的新 `DURATION` 小于旧的 `SHARD DURATION` ，并且TSDB For InfluxDB®正在将数据写入一个旧的、 `DURATION` 较长的shard group，那么系统将强制把所有数据存储在该shard group中，即使该shard group中有些数据已经在新的 `DURATION` 之外。一旦shard group中所有数据都在新的 `DURATION` 之外，TSDB For InfluxDB®将会删除整个shard group，然后系统开始将数据写入具有新的、更短 `SHARD DURATION` 的shard group，避免进一步意想不到的数据保留。

为什么TSDB For InfluxDB®无法解析微秒单位？

在不同场景下：写入、查询以及在TSDB For InfluxDB®命令行界面（CLI）设置精度，用于指定微秒时间单位的语法也不同。下表显示了每个类别支持的语法：

	使用HTTP API写入数据	所有查询	在CLI设置精度
u	√	√	√
us			
μ		√	

	使用HTTP API写入数据	所有查询	在CLI设置精度
µs	□	□	□

如何使TSDB For InfluxDB®的CLI返回用户可读的时间戳？

在您首次连接CLI时，请指定rfc3339精度：

```
$ influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242-precision rfc3339
```

或者，在连接CLI后指定精度：

```
$ influx -ssl -username <账号名称>-password <密码>-host <网络地址>-port 3242
Connected to https://<网络地址>:3242 version 1.7.x
> precision rfc3339
>
```

请查看文档命令行界面了解更多有用的CLI选项。

非admin用户如何使用 `USE` 指定一个数据库？

如果非admin用户拥有数据库的 `READ` 和/或 `WRITE` 权限，那么可以执行 `USE <database_name>` 语句。

如果非admin用户尝试用 `USE` 指定一个他们没有 `READ` 和/或 `WRITE` 权限的数据库，那么系统会返回错误：

```
ERR:Database<database_name> doesn't exist. Run SHOW DATABASES for a list of existing databases.
```

> 说明
>
> SHOW DATABASES查询只返回那些非admin用户有READ或WRITE权限的数据库。

如何使用TSDB For InfluxDB®的CLI将数据写入一个非默认的保留策略

请使用语法 `INSERT INTO [<database>.]<retention_policy> <line_protocol>` 将数据写入一个非默认的保留策略。（只允许在CLI中使用这种方式指定数据库和保留策略。如果通过HTTP写入数据，必须使用参数 `db` 和 `rp` 分别指定数据库和保留策略，指定保留策略是可选的。）

示例：

```
> INSERT INTO one_day mortality bool=true
Using retention policy one_day
> SELECT * FROM "mydb"."one_day"."mortality"
name: mortality
---------------
time                            bool
2016-09-13T22:29:43.229530864Z   true
```

请注意，您需要完全限定measurement来查询非默认保留策略中的数据。使用以下语法完全限定measurement：

```
"<database>"."<retention_policy>"."<measurement>"
```

为什么不能查询布尔类型的field value?

写入和查询布尔值的语法不一样。

布尔值语法	写入	查询
`t`，`f`	√	□
`T`，`F`	√	□
`true`，`false`	√	√
`True`，`False`	√	√
`TRUE`，`FALSE`	√	√

例如，`SELECT * FROM "hamlet" WHERE "bool"=True` 返回所有 `bool` 等于 `TRUE` 的数据点，但是，`SELECT * FROM "hamlet" WHERE "bool"=T` 不会返回任何结果。

TSDB For InfluxDB®如何处理多个shard之间的field的类型差异?

field value可以是浮点数、整数、字符串或者布尔值。在一个shard里面，field value的数据类型不能不一样，但是在不同的shard，field value的数据类型可以不同。

SELECT语句

`SELECT` 语句返回所有的field value**如果**这些值有相同的数据类型。如果在不同的shard里面field value的数据类型不一样，那么TSDB For InfluxDB®首先会执行类型转换（如果适用的话），然后返回所有值，并且按以下数据类型的顺序返回结果：浮点数，整数，字符串，布尔值。

如果在您的数据中，field value的类型不同，请使用语法 `<field_key>::<type>` 查询不同的数据类型。

示例

measurement `just_my_type` 有一个名为 `my_field` 的field， `my_field` 在四个不同的shard中有四个field value，并且每个field value的数据类型不一样（分别是浮点数、整数、字符串和布尔值）。

`SELECT *` 只返回浮点型和整型的field value。请注意，在返回结果中TSDB For InfluxDB®强制将整数转换成浮点数。

```
> SELECT * FROM just_my_type

name: just_my_type
------------------
time                        my_field
2016-06-03T15:45:00Z9.87034
2016-06-03T16:45:00Z7
```

`SELECT <field_key>::<type> [...]` 返回所有数据类型。TSDB For InfluxDB®将每种类型的数据输出到单独的列中，并且使用递增的列名表示。在可能的情况下，TSDB For InfluxDB®将field value转换成另一种数据类型；它将整数 `7` 转换成第一列中的浮点数，将浮点数 `9.879034` 转换成第二列中的整数。TSDB For InfluxDB®不能将浮点数或整数转换成字符串或布尔值。

```
> SELECT "my_field"::float,"my_field"::integer,"my_field"::string,"my_field"::boolean FROM
just_my_type

name: just_my_type
------------------
time                    my_field  my_field_1  my_field_2  my_field_3
2016-06-03T15:45:00Z9.870349
2016-06-03T16:45:00Z77
2016-06-03T17:45:00Z                          a string
2016-06-03T18:45:00Z                                      true
```

SHOW FIELD KEYS查询

`SHOW FIELD KEYS` 返回field key对应的每个shard中的每种数据类型。

示例

measurement `just_my_type` 有一个名为 `my_field` 的field， `my_field` 在四个不同的shard中有四个field value，并且每个field value的数据类型不一样（分别是浮点数、整数、字符串和布尔值）。

`SHOW FIELD KEYS` 返回所有四种数据类型：

```
> SHOW FIELD KEYS

name: just_my_type
fieldKey   fieldType
----------------
my_field   float
my_field   string
my_field   integer
my_field   boolean
```

TSDB For InfluxDB®可以存储的最小和最大整数是多少？

TSDB For InfluxDB®将所有整数存储为有符号的int64数据类型。int64有效的最小和最大值分别是 `-9223372036854775808` 和 `9223372036854775807` 。请查看Go builtins获得更多相关信息。

使用接近最小/最大整数但依旧在限制范围内的值可能会导致非预期的结果；有些函数和运算符会在计算过程中将数据类型int64转换成float64，这会引起溢出问题。

TSDB For InfluxDB®可以存储的最小和最大时间戳是多少？

最小的时间戳是 `-9223372036854775806` 或 `1677-09-21T00:12:43.145224194Z` ，最大的时间戳是 `9223372036854775806` 或 `2262-04-11T23:47:16.854775806Z` 。超出该范围的时间戳会返回解析错误。

如何知道存储在field中的数据类型？

`SHOW FIELD KEYS` 查询还返回field的数据类型。

示例

```
> SHOW FIELD KEYS FROM all_the_types
name: all_the_types
------------------
fieldKey  fieldType
blue      string
green     boolean
orange    integer
yellow    float
```

是否可以改变field的数据类型？

目前，在改变field的数据类型上面，TSDB For InfluxDB®提供非常有限的支持。语法 `<field_key>::<type>` 支持将field value从整数转换为浮点数或者从浮点数转换为整数。请查看文档**数据探索**获得更多关于转换操作的信息。无法将浮点数或整数转换为字符串或布尔值（反之亦然）。

我们列出了可用于更改field数据类型的方法：

将数据写入一个不同的field

最简单的解决方法就是将具有新数据类型的数据写入到同一个序列中的不同field。

使用shard系统

在一个shard里面，field value的数据类型不能不一样，但是在不同的shard，field value的数据类型可以不同。

如果要更改field的数据类型，用户可以使用 `SHOW SHARDS` 查询来识别当前shard的 `end_time` 。如果数据点的时间戳发生在 `end_time` 之后，那么TSDB For InfluxDB®允许将不同数据类型的数据写入到一个现有的field中（例如，field原来接受整数，但是在 `end_time` 之后，该field可以接受浮点数）。

请注意，这不会改变原来shard里面field的数据类型。

如何执行函数中的数学运算？

目前，TSDB For InfluxDB®不支持函数内的数学运算。我们建议使用子查询作为解决方法。

示例

InfluxQL不支持以下语法：

```
SELECT MEAN("dogs"-"cats")from"pet_daycare"
```

相反，我们可以使用子查询获得相同的结果：

```
> SELECT MEAN("difference") FROM (SELECT "dogs"-"cat" AS "difference" FROM "pet_daycare")
```

请查看文档**数据探索**获得更多关于子查询的信息。

为什么查询将epoch 0作为时间戳返回？

在TSDB For InfluxDB®中，epoch 0（ `1970-01-01T00:00:00Z` ）通常用作空时间戳（null timestamp），如果您请求的查询中没有时间戳返回，例如，对于没有规定时间范围的聚合函数，TSDB For InfluxDB®返回epoch 0作为时间戳。

哪些InfluxQL函数支持嵌套使用？

以下InfluxQL函数支持嵌套使用：

- `COUNT()` 嵌套 `DISTINCT()`
- `CUMULATIVE_SUM()`
- `DERIVATIVE()`
- `DIFFERENCE()`
- `ELAPSED()`
- `MOVING_AVERAGE()`
- `NON_NEGATIVE_DERIVATIVE()`
- `HOLT_WINTERS()` 和 `HOLT_WINTERS_WITH_FIT()`

关于如何使用子查询代替嵌套函数，请查看文档**数据探索**。

什么决定了 `GROUP BY time()` 查询返回的时间间隔？

`GROUP BY time()` 查询返回的时间间隔符合TSDB For InfluxDB®的预设时间段或者符合用户指定的偏移间隔。

示例

预设时间段

以下查询计算 `sunflowers` 在6:15pm到7:45pm之间的平均值，并将这些平均值按一小时进行分组：

```
SELECT mean("sunflowers")
FROM "flower_orders"
WHERE time >='2016-08-29T18:15:00Z' AND time <='2016-08-29T19:45:00Z' GROUP BY time(1h)
```

下面的结果展示了TSDB For InfluxDB®如何维护它的预设时间段。

在这个示例中，6pm是一个预设的时间段，7pm也是一个预设的时间段。由于 `WHERE` 子句中指定了查询的时间范围，所以在计算6pm时间段对应的平均值时不包括在6:15pm之前的数据，但是用于计算6pm时间段平均值的数据必须发生在6pm这个小时里。对于7pm时间段也是一样；用于计算7pm时间段平均值的数据必须发生在7pm这个小时里。虚线部分展示了用于计算每个平均值的数据点。

请注意，虽然结果中第一个时间戳是 `2016-08-29T18:00:00Z` ，但是在该时间段的查询结果不包含发生在 `2016-08-29T18:15:00Z` （ `WHERE` 子句中指定的开始时间）之前的数据。

原始数据： 结果：

```
name: flower_orders                              name: flower_orders
___________________________
time                    sunflowers               time                  mean
2016-08-29T18:00:00Z342016-08-29T18:00:00Z22.332
|--|2016-08-29T19:00:00Z62.75
2016-08-29T18:15:00Z|28|
2016-08-29T18:30:00Z|19|
2016-08-29T18:45:00Z|20|
|--|
|--|
2016-08-29T19:00:00Z|56|
2016-08-29T19:15:00Z|76|
2016-08-29T19:30:00Z|29|
2016-08-29T19:45:00Z|90|
|--|
2016-08-29T20:00:00Z70
```

偏移间隔

以下查询计算 `sunflowers` 在6:15pm到7:45pm之间的平均值，并将这些平均值按一小时进行分组，同时，该查询还将TSDB For InfluxDB®的预设时间段偏移 `15` 分钟：

```
SELECT mean("sunflowers")
FROM "flower_orders"
WHERE time >='2016-08-29T18:15:00Z' AND time <='2016-08-29T19:45:00Z' GROUP BY time(1h,15m)
---
|
                                                                                offset in
terval
```

在这个示例中，用户指定的偏移间隔将TSDB For InfluxDB®的预设时间段前移了 `15` 分钟。现在，6pm时间段的平均值包括在6:15pm和7:15pm之间的数据，7pm时间段的平均值包括在7:15pm和8:15pm之间的数据。虚线部分展示了用于计算每个平均值的数据点。

请注意，现在结果中第一个时间戳是 `2016-08-29T18:15:00Z` ，而不是 `2016-08-29T18:00:00Z` 。

原始数据： 结果：

```
name: flower_orders                              name: flower_orders
———————————————————————                          ——————————————————
time                    sunflowers               time                  mean
2016-08-29T18:00:00Z342016-08-29T18:15:00Z30.75
|--|2016-08-29T19:15:00Z65
2016-08-29T18:15:00Z|28|
2016-08-29T18:30:00Z|19|
2016-08-29T18:45:00Z|20|
2016-08-29T19:00:00Z|56|
|--|
|--|
2016-08-29T19:15:00Z|76|
2016-08-29T19:30:00Z|29|
2016-08-29T19:45:00Z|90|
2016-08-29T20:00:00Z|70|
|--|
```

为什么查询没有返回任何数据或者只返回一部分数据？

对于为什么查询没有返回任何数据或者只返回一部分数据，有几种可能的解释。我们在下面列出了一些最常见的原因：

保留策略

第一个也是最常见的解释与保留策略（RP）有关。TSDB For InfluxDB®自动从数据库的默认（ `DEFAULT` ）RP中查询数据。如果您的数据不是存储在默认的RP，TSDB For InfluxDB®不会返回任何结果，除非您明确指定所使用的RP。

SELECT子句中的tag key

在 `SELECT` 子句中，至少需要包含一个field key，查询才会返回数据。如果 `SELECT` 子句只包含一个或多个tag key，查询会返回空的结果。请查看文档**数据探索**获得更多相关信息。

查询时间范围

另一个可能的解释与查询的时间范围有关。默认情况下，大多数 `SELECT` 查询涵盖在 `1677-09-21 00:12:43.145224194` UTC和 `2262-04-11T23:47:16.854775806Z` UTC之间的时间范围。`SELECT` 查询还包括 `GROUP BY time()` 子句，但是，它涵盖的时间范围在 `1677-09-21 00:12:43.145224194` 和 `now()` 之间。如果您的数据发生在 `now()` 之后，那么 `GROUP BY time()` 查询不会覆盖这些发生在 `now()` 之后的数据。如果查询语句包括 `GROUP BY time()` 子句，并且有数据发生在 `now()` 之后，您需要为时间范围提供一个上限。

标识符名字

最后一个常见的解释与schema有关（field和tag有相同的名字）。如果field key和tag key相同，那么在所有查询中优先考虑field。在查询中，你需要使用 `::tag` 语法指定tag key。

为什么 `GROUP BY time()` 查询不返回发生在 `now()` 之后的时间戳？

大多数 `SELECT` 语句的默认时间范围在 `1677-09-21 00:12:43.145224194 UTC` 和 `2262-04-11T23:47:16.854775806Z UTC` 之间。对于带 `GROUP BY time()` 子句的 `SELECT` 语句，默认的时间范围在 `1677-09-21 00:12:43.145224194` 和 `now()` 之间。

如果要查询时间戳发生在 `now()` 之后的数据，带 `GROUP BY time()` 子句的 `SELECT` 语句必须在 `WHERE` 子句中提供一个时间上限。

在下面的示例中，第一个查询涵盖时间戳在 `2015-09-18T21:30:00Z` 和 `now()` 之间的数据，第二个查询涵盖时间戳在 `2015-09-18T21:30:00Z` 和 `now()` 之后180个星期之间的数据。

```
> SELECT MEAN("boards") FROM "hillvalley" WHERE time >='2015-09-18T21:30:00Z' GROUP BY time
(12m) fill(none)


> SELECT MEAN("boards") FROM "hillvalley" WHERE time >='2015-09-18T21:30:00Z' AND time <= n
ow()+180w GROUP BY time(12m) fill(none)
```

请注意， `WHERE` 子句必须提供一个时间上线来覆盖默认的 `now()` 上限。下面的查询只是将 `now()` 的下限重置，使得查询的时间范围在 `now()` 和 `now()` 之间：

```
> SELECT MEAN("boards") FROM "hillvalley" WHERE time >= now() GROUP BY time(12m) fill(none)
>
```

请查看文档**数据探索**获得更多关于时间语法的信息。

是否可以对时间戳执行数学运算？

目前，在TSDB For InfluxDB®中，不能对时间戳执行数学运算。更多关于时间的计算必须由接收查询结果的客户端执行。

对时间戳使用InfluxQL函数，TSDB For InfluxDB®仅提供有限的支持。ELAPSED()函数返回单个field中时间戳之间的差值。

是否可以从返回的时间戳中识别写入精度？

不管提供的写入精度是多少，TSDB For InfluxDB®将所有时间戳存储为纳秒。需要注意的一个重要事项是，当返回查询结果时，数据库会不动声色地删除时间戳后面的零，使原始的写入精度很难识别。

在下面的示例中，tag `precision_supplied` 和 `timestamp_supplied` 分别显示了用户在写入数据时提供的时间精度和时间戳。因为TSDB For InfluxDB®默默地将返回的时间戳后面的零删除了，所以从返回的时间戳中很难识别写入精度。

```
name: trails
-------------
time                  value  precision_supplied  timestamp_supplied
1970-01-01T01:00:00Z3      n                   3600000000000
1970-01-01T01:00:00Z5      h                   1
1970-01-01T02:00:00Z4      n                   7200000000000
1970-01-01T02:00:00Z6      h                   2
```

当查询数据时，什么时候应该使用单引号，什么时候应该使用双引号？

用单引号将字符串类型的值括起来（例如，tag value），但是不要用单引号将标识符（数据库名字、保留策略名字、用户名、measurement的名字、tag key和field key）括起来。

如果标识符以数字开头，或包含除 `[A-z,0-9,_]` 外的字符，或者标识符是InfluxQL关键字，那么需要使用双引号将标识符括起来。如果标识符不属于这些类别之一，可以不需要使用双引号将它们括起来，但是我们还是建议用双引号将它们括起来。

示例：

合法的查询：`SELECT bikes_available FROM bikes WHERE station_id='9'`

合法的查询：`SELECT "bikes_available" FROM "bikes" WHERE "station_id"='9'`

合法的查询：`SELECT MIN("avgrq-sz") AS "min_avgrq-sz" FROM telegraf`

合法的查询：`SELECT * from "cr@zy" where "p^e"='2'`

非法的查询：`SELECT 'bikes_available' FROM 'bikes' WHERE 'station_id'="9"`

非法的查询：`SELECT * from cr@zy where p^e='2'`

用单引号将日期时间字符串括起来。如果您使用双引号将日期时间字符串括起来，TSDB For InfluxDB®会返回错误（ `ERR: invalid operation: time and *influxql.VarRef are not compatible` ）。

示例：

合法的查询：

```
 SELECT "water_level" FROM "h2o_feet" WHERE time > '2015-08-18T23:00:01.232000000Z' AND time
< '2015-09-19'
```

非法的查询：

```
SELECT "water_level" FROM "h2o_feet" WHERE time > "2015-08-18T23:00:01.232000000Z" AND time
< "2015-09-19"
```

请查看文档**数据探索**获得更多关于时间语法的信息。

为什么在创建一个新的默认（ `DEFAULT` ）保留策略后会丢失数据？

当您在数据库中创建一个新的默认保留策略（RP）后，在旧的默认RP中的数据依旧保存在旧的RP中。对于不指定RP的查询，将会自动查询新默认RP中的数据，所有旧数据可能会丢失。为了查询旧数据，必须完全限定查询中的数据。

示例：

在measurement `fleeting` 中的所有数据属于默认的RP，该RP的名字为 `one_hour` ：

```
> SELECT count(flounders) FROM fleeting
name: fleeting
--------------
time                    count
1970-01-01T00:00:00Z8
```

现在我们创建一个新的默认RP（ `two_hour` ），并执行相同的查询：

```
> SELECT count(flounders) FROM fleeting
>
```

为了查询旧数据，我们必须通过完全限定 `fleeting` 来指定旧的默认RP：

```
> SELECT count(flounders) FROM fish.one_hour.fleeting
name: fleeting
--------------
time                    count
1970-01-01T00:00:00Z8
```

为什么带有 `WHERE OR` 时间子句的查询返回空结果？

目前，TSDB For InfluxDB®不支持在 `WHERE` 子句中使用 `OR` 来指定多个时间范围。如果查询中的 `WHERE` 子句使用 `OR` 来指定多个时间范围，那么TSDB For InfluxDB®不会返回任何结果。

示例：

```
> SELECT * FROM "absolutismus" WHERE time ='2016-07-31T20:07:00Z' OR time ='2016-07-31T23:0
7:17Z'
>
```

为什么 `fill(previous)` 返回空结果？

如果前一个值在查询的时间范围之外，那么 `fill(previous)` 不会填充该时间段的值。

在下面的示例中，TSDB For InfluxDB®不会使用时间段 `2016-07-12T16:50:00Z` - `2016-07-12T16:50:10Z` 的值填充时间段 `2016-07-12T16:50:20Z` - `2016-07-12T16:50:30Z` ，因为该查询的时间范围并不包含较早的时间段。

原始数据：

```
> SELECT * FROM "cupcakes"
name: cupcakes
--------------
time                    chocolate
2016-07-12T16:50:00Z3
2016-07-12T16:50:10Z2
2016-07-12T16:50:40Z12
2016-07-12T16:50:50Z11
```

`GROUP BY time()` 查询：

```
> SELECT max("chocolate") FROM "cupcakes" WHERE time >='2016-07-12T16:50:20Z' AND time <='2
016-07-12T16:51:10Z' GROUP BY time(20s) fill(previous)
name: cupcakes
--------------
time                    max
2016-07-12T16:50:20Z
2016-07-12T16:50:40Z12
2016-07-12T16:51:00Z12
```

为什么 `INTO` 查询会丢失数据？

默认情况下， `INTO` 查询将原始数据中的tag转换成新写入数据的field。这会导致TSDB For InfluxDB®覆盖之前由tag区分的数据点。在所有 `INTO` 查询中加上 `GROUP BY *` ，可以将tag保留在新写入的数据中。

请注意，这种方式不适用于使用 `TOP()` 或 `BOTTOM()` 函数的查询。请查看文档InfluxDB函数获得更多关于 `TOP()` 和 `BOTTOM()` 的信息。

示例

原始数据

measurement `french_bulldogs` 包含一个tag `color` 和一个field `name` 。

```
> SELECT * FROM "french_bulldogs"
name: french_bulldogs
---------------------
time                  color  name
2016-05-25T00:05:00Z  peach  nugget
2016-05-25T00:05:00Z  grey   rumple
2016-05-25T00:10:00Z  black  prince
```

不使用GROUP BY *的INTO查询

不使用 `GROUP BY *` 子句的 `INTO` 查询将tag `color` 转换成新写入数据中的field。在原始数据中，数据点 `nugget` 和 `rumple` 仅由tag `color` 区分。一旦 `color` 变成field，TSDB For InfluxDB®会认为数据点 `nugget` 和 `rumple` 是重复的，它会用数据点 `rumple` 将数据点 `nugget` 覆盖。

```
> SELECT * INTO "all_dogs" FROM "french_bulldogs"
name: result
------------
time                  written
1970-01-01T00:00:00Z3

> SELECT * FROM "all_dogs"
name: all_dogs
--------------
time                  color  name
2016-05-25T00:05:00Z  grey   rumple               <---- no more nugget
2016-05-25T00:10:00Z  black  prince
```

使用GROUP BY *的INTO查询

使用 `GROUP BY *` 子句的 `INTO` 查询将tag `color` 保留在新写入的数据中。在这种情况下，数据点 `nugget` 和 `rumple` 依旧是不同的数据点，TSDB For InfluxDB®不会覆盖任何数据。

```
> SELECT "name" INTO "all_dogs" FROM "french_bulldogs" GROUP BY *
name: result
------------
time                  written
1970-01-01T00:00:00Z3

> SELECT * FROM "all_dogs"
name: all_dogs
--------------
time                  color  name
2016-05-25T00:05:00Z  peach  nugget
2016-05-25T00:05:00Z  grey   rumple
2016-05-25T00:10:00Z  black  prince
```

如何查询tag key和field key名字相同的数据？

使用语法 `::` 指定一个key是field key还是tag key。

示例

示例数据：

```
> INSERT candied,almonds=true almonds=50,half_almonds=511465317610000000000
> INSERT candied,almonds=true almonds=55,half_almonds=561465317620000000000

> SELECT * FROM "candied"
name: candied
-------------
time                    almonds  almonds_1  half_almonds
2016-06-07T16:40:10Z50       true       51
2016-06-07T16:40:20Z55       true       56
```

指定key是field：

```
> SELECT * FROM "candied" WHERE "almonds"::field >51
name: candied
-------------
time                    almonds  almonds_1  half_almonds
2016-06-07T16:40:20Z55       true       56
```

指定key是tag：

```
> SELECT * FROM "candied" WHERE "almonds"::tag='true'
name: candied
-------------
time                    almonds  almonds_1  half_almonds
2016-06-07T16:40:10Z50       true       51
2016-06-07T16:40:20Z55       true       56
```

如何跨measurement查询数据？

目前，无法跨measurement执行数学运算或分组。所有数据必须在同一个measurement下，才能一起查询这些数据。TSDB For InfluxDB®不是一个关系型数据库，跨measurement映射数据目前不是一个推荐的schema。

时间戳的顺序是否重要？

不重要。测试结果表明TSDB For InfluxDB®完成以下查询所需的时间差别非常小：

```
SELECT ... FROM ... WHERE time >'timestamp1' AND time <'timestamp2'
SELECT ... FROM ... WHERE time <'timestamp2' AND time >'timestamp1'
```

如何 `SELECT` 有tag但没有tag value的数据？

使用 `''` 指定一个空的tag value。例如：

```
> SELECT * FROM "vases" WHERE priceless=''
name: vases
-----------
time                  origin   priceless
2016-07-20T18:42:00Z8
```

为什么序列基数很重要?

TSDB For InfluxDB®维护系统中每个序列在内存中的索引。随着序列数量不断增加，RAM（内存）使用量也在不断增加。序列基数过大会导致操作系统终止TSDB For InfluxDB®进程，并抛出内存不足（OOM）异常。请查看文档InfluxQL参考了解关于序列基数的InfluxQL命令。

如何写入整型的field value?

当写入整数时，在field value末尾加上 `i` 。如果您不加上 `i` ，TSDB For InfluxDB®会把field value当作浮点数。

写入整数： `value=100i` 写入浮点数： `value=100`

TSDB For InfluxDB®如何处理重复数据点?

measurement的名字、tag set和时间戳唯一标识一个数据点。如果您提交的数据点跟已有的数据点相比，具有相同measurement、tag set和时间戳，但具有不同field set，那么该数据点的field set会变为旧field set和新field set的并集，如果有任何冲突以新field set为准。这是预期的结果。

例如：

旧数据点： `cpu_load,hostname=server02,az=us_west val_1=24.5,val_2=7 1234567890000000`

新数据点： `cpu_load,hostname=server02,az=us_west val_1=5.24 1234567890000000`

当您提交新数据点后，TSDB For InfluxDB®使用新的field value覆盖 `val_1` 的值， `val_2` 的值继续保留：

```
> SELECT * FROM "cpu_load" WHERE time =1234567890000000
name: cpu_load
--------------
time                     az        hostname   val_1   val_2
1970-01-15T06:56:07.89Z  us_west   server02   5.247
```

为了存储这两个数据点，可以：

- 引入新的tag保证唯一性。

  旧数据点：

  `cpu_load,hostname=server02,az=us_west,uniq=1 val_1=24.5,val_2=7 1234567890000000`

  新数据点： `cpu_load,hostname=server02,az=us_west,uniq=2 val_1=5.24 1234567890000000`

  将新数据点写入TSDB For InfluxDB®后：

```
> SELECT * FROM "cpu_load" WHERE time =1234567890000000
name: cpu_load
--------------
time                        az        hostname   uniq   val_1   val_2
1970-01-15T06:56:07.89Z     us_west   server02   124.57
1970-01-15T06:56:07.89Z     us_west   server02   25.24
```

- 时间戳增加一纳秒。

  旧数据点：`cpu_load,hostname=server02,az=us_west val_1=24.5,val_2=7 1234567890000000`

  新数据点：`cpu_load,hostname=server02,az=us_west val_1=5.24 1234567890000001`

  将新数据点写入TSDB For InfluxDB®后：

```
> SELECT * FROM "cpu_load" WHERE time >=1234567890000000 and time <=1234567890000001
name: cpu_load
--------------
time                             az        hostname   val_1   val_2
1970-01-15T06:56:07.89Z          us_west   server02   24.57
1970-01-15T06:56:07.8900000001Z  us_west   server02   5.24
```

HTTP API需要怎样的换行符？

TSDB For InfluxDB®的行协议依赖换行符（`\n`，这是ASCII `0x0A`）来表示一行的结束和新的一行的开始。文件或数据使用 `\n` 以外的换行符会导致以下错误：`bad timestamp`，`unable to parse`。

请注意，Windows使用回车键和换行符（`\r\n`）作为换行符。

当将数据写入TSDB For InfluxDB®时，应该避免哪些文字和字符？

InfluxQL关键字

如果您使用InfluxQL关键字作为标识符，您需要在每个查询中使用双引号将该标识符括起来。如果不使用双引号，会导致错误。标识符是连续查询名字、数据库名字、field key、measurement的名字、保留策略名字、tag key和用户名。

时间

关键字 `time` 是一个特例。`time` 可以是一个连续查询名字、数据库名字、measurement的名字、保留策略名字和用户名。在这些情况下，不需要在查询中用双引号将 `time` 括起来。`time` 不能是field key或tag key；TSDB For InfluxDB®拒绝写入将 `time` 作为field key或tag key的数据，对于这种数据写入，TSDB For InfluxDB®会返回错误。

示例

将time作为measurement，写入数据并查询它

```
> INSERT time value=1

> SELECT * FROM time

name: time
time                           value
---------
2017-02-07T18:28:27.34978538４Z1
```

在TSDB For InfluxDB®中，`time`是一个有效的measurement名字。

将time作为field key，写入数据并尝试查询它

```
> INSERT mymeas time=1
ERR:{"error":"partial write: invalid field name: input field \"time\" on measurement \"myme
as\" is invalid dropped=1"}
```

在TSDB For InfluxDB®中，`time`不是一个有效的field key。系统无法写入该数据点，并且返回`400`错误。

将time作为tag key，写入数据并尝试查询它

```
> INSERT mymeas,time=1 value=1
ERR:{"error":"partial write: invalid tag key: input tag \"time\" on measurement \"mymeas\"
is invalid dropped=1"}
```

在TSDB For InfluxDB®中，`time`不是一个有效的tag key。系统无法写入该数据点，并且返回`400`错误。

字符

为了保持简单的正则表达式和引号，避免在标识符中使用以下字符：`\`反斜杠`^`尖号`$`货币符号`'`单引号`"`双引号`=`等号`,`逗号

当写入数据时，什么时候应该使用单引号，什么时候应该使用双引号？

- 通过行协议写入数据时，避免使用单引号和双引号将标识符括起来；请查看下面的示例，使用引号后的标识符会使查询变得复杂。标识符是连续查询名字、数据库名字、field key、measurement的名字、保留策略名字、subscription的名字、tag key和用户名。

  写入带双引号的measurement：`INSERT "bikes" bikes_available=3`适用的查询：

  `SELECT * FROM "\"bikes\""`

  写入带单引号的measurement：`INSERT 'bikes' bikes_available=3`适用的查询：

  `SELECT * FROM "\'bikes\'"`

写入不带引号的measurement：`INSERT bikes bikes_available=3` 适用的查询：

```
SELECT * FROM "bikes"
```

- 用双引号将字符串类型的field value括起来。

  写入：`INSERT bikes happiness="level 2"` 适用的查询：

  ```
  SELECT * FROM "bikes" WHERE "happiness"='level 2'
  ```

- 应该用反斜杠转义特殊字符，而不是用引号将其括起来。

  写入：`INSERT wacky va\"ue=4` 适用的查询：`SELECT "va\"ue" FROM "wacky"`

请查看文档行协议参考获得更多相关信息。

时间戳的精度是否重要？

重要。为了最大限度地提高性能，向TSDB For InfluxDB®写入数据时尽量使用最粗糙的时间精度。

在下面两个例子中，第一个请求使用默认精度（纳秒），而第二个请求将精度设置为秒：

```
curl -i -XPOST "https://<网络地址>:3242/write?db=weather&u=<账号名称>&p=<密码>"--data-binary '
temperature,location=1 value=90 1472666050000000000'

curl -i -XPOST "https://<网络地址>:3242/write?db=weather&precision=s&u=<账号名称>&p=<密码>"--d
ata-binary 'temperature,location=1 value=90 1472666050'
```

虽然性能会提高，但是代价是精度越粗糙，越有可能出现具有相同时间戳的重复数据点，可能会覆盖其它数据点。

14.云实例监控

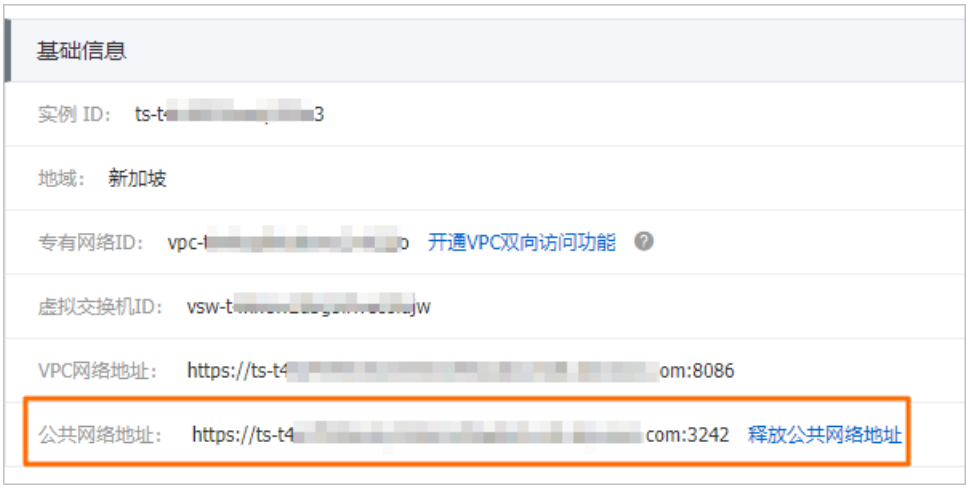
本文将介绍如何使用阿里云InfluxDB®实例的云实例监控服务。

背景信息

用户在阿里云上购买的实例，例如MySQL、Redis、MongoDB，云实例监控为其提供了快速监控部署服务。目前支持针对阿里云InfluxDB®实例所在相同VPC内的MySQL、Redis、MongoDB类型云实例进行监控部署。

前提条件

- InfluxDB实例版本在1.7.11（含）以上的版本，低版本可以提交工单进行升级之后使用。
- 已开通VPC双向访问功能，详情请参见VPC双向访问。
- 已将双向VPC访问白名单中的IP加入待监控的云实例的白名单中，详情请参见设置网络白名单。
- 已开通实例公网访问，如下图所示。



- 开通了grafana公网访问链接，并创建了相应的账号。

操作步骤

1. 登录TSDB控制台。
2. 在**实例列表**界面，选择目标实例操作列的**管理**。
3. 在左侧导航栏，单击**云实例监控**。
4. 创建监控项。

i. 单击**创建实例监控**，设置如下参数。

参数	说明
选择引擎	选择引擎类型。 目前支持MySQL、Redis、MongoDB。
实例ID	选择实例ID
授权账号和授权密码	输入用户名和密码，例如：MySQL用户名和密码。

ii. 单击**验证信息和授权**，对监控项能否通过授权账号和密码进行连接并验证。

iii. 设置存储数据库参数。

参数	说明
存储数据库	InfluxDB的数据库名。
用户名和密码	InfluxDB的用户名和密码。

iv. 单击**写入数据库验证**，将自动进行InfluxDB的数据库的用户名和密码验证。

> **说明**
> 如果用户存在，将验证是否对指定的数据库具备读写权限。如果用户不存在，将自动创建用户。

v. 执行创建。

> **说明**
> 如果失败，会提示可以重试。如果重试多次依然失败，请提交工单进行处理。

5. 单击**修改**，修改监控项基本信息。
6. 单击**查看监控**，进入实例对应的Grafana监控页面。

> **说明**
> 第一次使用需要登录用户名和密码。对应的Grafana的用户名和密码需在控制台的可视化Grafana页面创建。

15.数据备份管理

数据备份功能支持用户对阿里云TSDB for InfluxDB®实例的数据库进行备份。数据备份后，您可以根据需要进行数据还原、删除操作。本文介绍如何创建数据备份任务、数据还原和删除数据备份任务。

前提条件

已购买TSDB实例并创建数据库。具体操作，请参见购买流程和管理用户账号和数据库。

操作步骤

1. 登录TSDB管理控制台。
2. 找到目标实例，单击**操作**列中的**管理**。
3. 在左侧导航栏，选择**时序数据管理 > 数据备份**。
4. 在**数据备份列表**页面，单击**创建备份**。
5. 在弹出的**创建备份**对话框中，配置以下参数。

参数	说明
备份数据库	选择需要备份的数据库。

参数	说明
任务名称	数据备份的任务名称。系统默认生成一个任务名称，您也可以按照需求自定义任务名称。任务名称要求如下： ○ 至少包含小写字母、数字、下划线、中划线其中任意一项。 ○ 字母开头，字母或数字结尾。 ○ 最长为64个字符。
全量备份	是否支持全量备份。默认开启，表示支持全量备份。您也可以关闭，然后自定义数据备份的**开始时间**和**结束时间**。

6. 配置完成后，单击**确定**。
7. （可选）数据还原。
   i. 选择需要还原的数据备份任务，单击**操作**列中的**还原**。

ii. 在弹出的**创建还原**对话框，配置以下参数，然后单击**确定**。

参数	说明
原数据库	需要进行数据还原的数据库。
任务名称	数据还原的任务名称。系统默认生成一个任务名称，您也可以按照需求自定义任务名称。任务名称要求如下： ■ 至少包含小写字母、数字、下划线、中划线其中任意一项。 ■ 字母开头，字母或数字结尾。 ■ 最长为64个字符。
导入到新的数据库	输入需要还原的数据库名称。 **说明** ■ 请确保此数据库当前还未在InfluxDB中创建过。 ■ 在数据还原未完成前，不允许进行创建数据库操作。

后续操作

- 数据备份任务创建完成后，您可以单击目标任务**操作**列中的**删除备份**，删除备份任务。
- 数据还原任务创建完成后，您可以单击目标任务**操作**列的**查看还原**，查看还原进度。