

ALIBABA CLOUD

阿里云

云数据库Cassandra版 最佳实践

文档版本：20210223

 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 <code>cd /d C:/window</code> 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	<code>bae log list --instanceid</code> <i>Instance_ID</i>
[] 或者 [a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ } 或者 {a b}	表示必选项，至多选择一个。	<code>switch {active stand}</code>

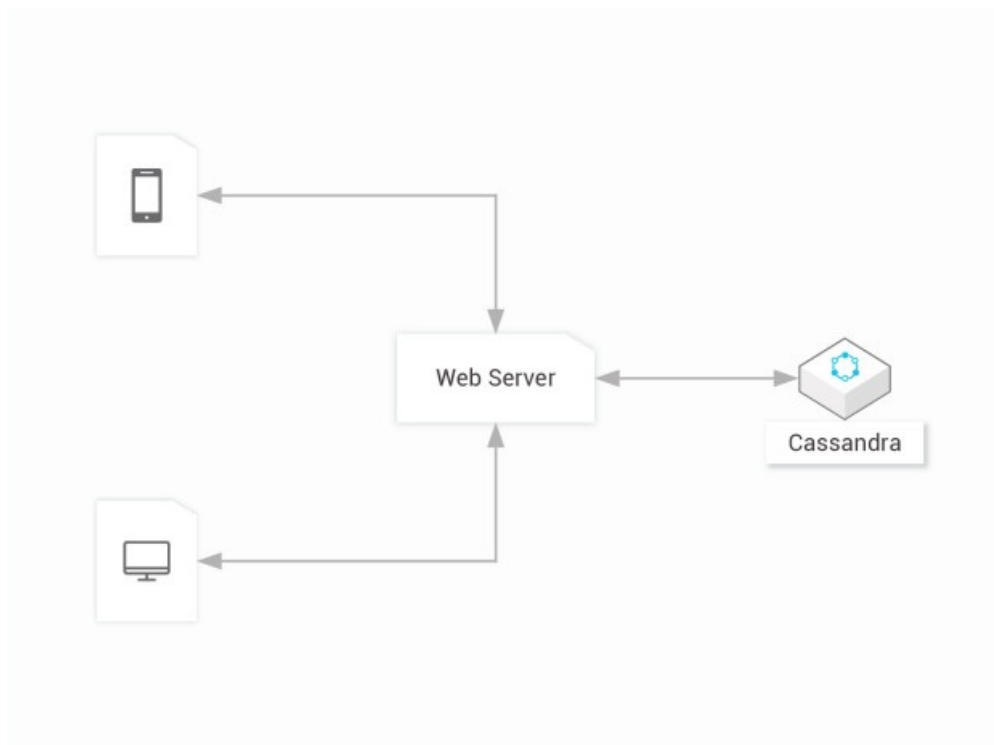
目录

1.云数据库Cassandra版应用场景	05
2.静态列(static column)实战	07

1.云数据库Cassandra版应用场景

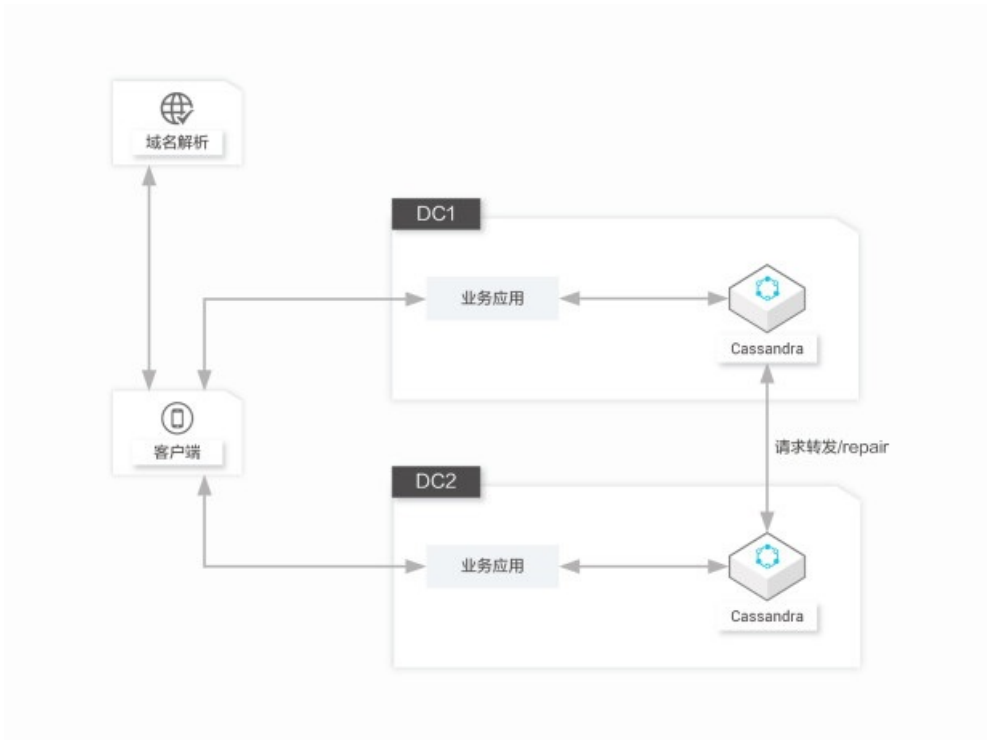
cassandra能够支持大并发低延时的访问需求，具备高可用和弹性扩容能力，适合消息、feed流、订单查询、网站等各种大数据量的互联网在线应用场景。

在线应用场景

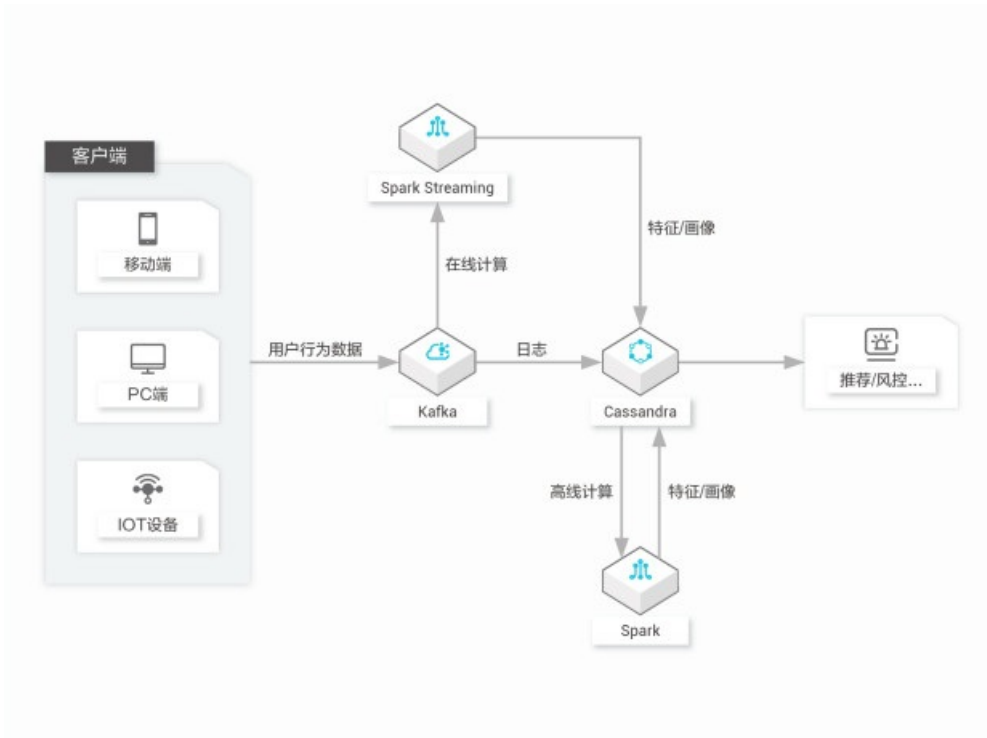


方案优势

- 高可用：单点故障不影响业务。
- 低延时：999延迟在毫秒级别。
- 弹性：可随着业务增长灵活扩容计算和存储能力。
- 多活：cassandra原生支持多DC部署方式。应用使用LOCAL_QUORUM的一致性级别访问cassandra，cassandra自动转发请求到另一个DC。实现更好的可用性和容灾能力。



- 容灾能力：避免物理灾害对数据和可用性造成影响。
- 实施简单：使用cassandra原生机制实现DC之间的数据同步，应用改动小。
- 数据驱动的业务-风控/推荐/IOT：多种数据源的用户行为数据实时存储在Cassandra中，并构建用户画像。提供风控、推荐等服务。



- 低成本存储：采用稀疏存储模式，适合用户画像高压压缩存储。
- 高扩展性：线性可扩展，灵活扩容计算和存储能力。
- 低延时：999延迟在毫秒级别。

2. 静态列(static column)实战

场景

需要 Cassandra 中使用一张表记录用户基本信息（比如 email、密码等）以及用户状态更新。通常来说，用户的基本信息一般很少会变动，但是用户状态会经常变化，如果每次状态更新都把用户基本信息都加进去，将浪费大量的存储空间。

为了解决这种问题，Cassandra 引入了 static column。同一个 partition key 中被声明为 static 的列只有一个值的，也就是只存储一份。

定义静态列

在表中将某个列定义为 STATIC 很简单，只需要在列的最后面加上 STATIC 关键字，具体如下：

```
CREATE TABLE "iteblog_users_with_status_updates" (  
  "username" text,  
  "id" timeuuid,  
  "email" text STATIC,  
  "encrypted_password" blob STATIC,  
  "body" text,  
  PRIMARY KEY ("username", "id")  
);
```

上述命令将表中的 email 和 encrypted_password 两个字段设置为 STATIC。这意味着同一个 username 只会有一个 email 和 encrypted_password。

静态列限制

不是任何表都支持为列加上 STATIC 关键字的，静态列有以下限制：

- 表没有定义 Clustering columns（又称 Clustering key），例如：

```
cqlsh:iteblog_keyspace> CREATE TABLE "iteblog_users_with_status_updates_invalid" (  
  ... "username" text,  
  ... "id" timeuuid,  
  ... "email" text STATIC,  
  ... "encrypted_password" blob STATIC,  
  ... "body" text,  
  ... PRIMARY KEY ("username")  
  ... );
```

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="Static columns are only useful (and thus allowed) if the table has at least one clustering column"
```

iteblog_users_with_status_updates_invalid 表只有 PRIMARY KEY，没有定义 clustering column，不支持创建 Static columns。这是因为静态列在同一个 partition key 存在多行的情况下才能达到最优情况，而且行数越多效果也好。但是如果没有定义 clustering column，相同 PRIMARY KEY 的数据在同一个分区里面只存在一行数据，本质上就是静态的，所以没必要支持静态列。

- 建表的时候指定了 COMPACT STORAGE，例如：

```
cqlsh:iteblog_keyspace> CREATE TABLE "iteblog_users_with_status_updates_invalid" (  
    ... "username" text,  
    ... "id" timeuuid,  
    ... "email" text STATIC,  
    ... "encrypted_password" blob STATIC,  
    ... "body" text,  
    ... PRIMARY KEY ("username", "id")  
    ... )WITH COMPACT STORAGE;
```

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="Static columns are not supported in  
COMPACT STORAGE tables"
```

- 列是 partition key/Clustering columns 的一部分，例如：

```
cqlsh:iteblog_keyspace> CREATE TABLE "iteblog_users_with_status_updates_invalid" (  
    ... "username" text,  
    ... "id" timeuuid STATIC,  
    ... "email" text STATIC,  
    ... "encrypted_password" blob STATIC,  
    ... "body" text,  
    ... PRIMARY KEY ("username", "id")  
    ... );
```

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="Static column id cannot be part of the  
PRIMARY KEY"
```

```
cqlsh:iteblog_keyspace> CREATE TABLE "iteblog_users_with_status_updates_invalid" (  
    ... "username" text,  
    ... "id" timeuuid,  
    ... "email" text STATIC,  
    ... "encrypted_password" blob STATIC,  
    ... "body" text,  
    ... PRIMARY KEY (("username", "id"), email)  
    ... );
```

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="Static column email cannot be part  
of the PRIMARY KEY"
```


为静态列的表插入数据

含有静态列的表插入数据和正常表类似，例如往 `iteblog_users_with_status_updates` 导入数据：

```
cqlsh:iteblog_keyspace> INSERT INTO "iteblog_users_with_status_updates"
... ("username", "id", "email", "encrypted_password", "body")
... VALUES (
... 'iteblog',
... NOW(),
... 'iteblog_hadoop@iteblog.com',
... 0x877E8C36EFA827DBD4CAFBC92DD90D76,
... 'Learning Cassandra!'
...);
cqlsh:iteblog_keyspace> select username, email, encrypted_password, body from iteblog_users_with_status_updates;
username | email          | encrypted_password          | body
-----+-----+-----+-----
iteblog | iteblog_hadoop@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76 | Learning Cassandra!
(1 rows)
```

可以看出，成功的插入一条数据了。上述语句做了两件事：

- 所有 `username` 为 `iteblog` 数据中的 `email` 和 `encrypted_password` 都被设置为 `iteblog_hadoop@iteblog.com` 和 `0x877e8c36efa827dbd4cafbc92dd90d76`。
- 在 `iteblog` 所在的分区中新增了 `body` 内容为 `Learning Cassandra!` 的记录。再往表中插入一条数据，如下：

```
cqlsh:iteblog_keyspace> INSERT INTO "iteblog_users_with_status_updates"
... ("username", "id", "body")
... VALUES ('iteblog', NOW(), 'I love Cassandra!');
cqlsh:iteblog_keyspace> select username, email, encrypted_password, body from iteblog_users_with_status_updates;
username | email          | encrypted_password          | body
-----+-----+-----+-----
iteblog | iteblog_hadoop@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76 | Learning Cassandra!
iteblog | iteblog_hadoop@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76 | I love Cassandra!
(2 rows)
cqlsh:iteblog_keyspace>
```

可以看出，这次插入数据的时候，并没有指定 `email` 和 `encrypted_password`。但是从查询结果可以看出，新增加的行 `email` 和 `encrypted_password` 的值和之前是一样的。

现在由于某些原因，用户修改了自己的 email，例如：

```
cqlsh:iteblog_keyspace> UPDATE iteblog_users_with_status_updates SET email = 'iteblog@iteblog.com'
... WHERE username = 'iteblog';
cqlsh:iteblog_keyspace> select username, email, encrypted_password, body from iteblog_users_with_status_updates;
username | email          | encrypted_password          | body
-----+-----+-----+-----
iteblog | iteblog@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76 | Learning Cassandra!
iteblog | iteblog@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76 | I love Cassandra!
(2 rows)
```

从上面查询这输出的结果可以看出，username 为 iteblog 的 email 全部修改成一样的了，这就是静态列的强大之处。

现在表中存在了用户的邮箱和密码等信息，如果在前端的页面支持用户修改自己的邮箱和密码，这时后台系统需要获取到现有的邮箱和密码，具体如下：

```
cqlsh:iteblog_keyspace> SELECT "username", "email", "encrypted_password"
... FROM "iteblog_users_with_status_updates"
... WHERE "username" = 'iteblog';
username | email          | encrypted_password
-----+-----+-----
iteblog | iteblog@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76
iteblog | iteblog@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76
(2 rows)
```

可以看出，表中有多少行 username 为 iteblog 的数据将会输出多少行邮箱和密码，这不是最终想要的数。此时您可以在查询的时候加上 DISTINCT 关键字，例如：

```
cqlsh:iteblog_keyspace> SELECT DISTINCT "username", "email", "encrypted_password"
... FROM "iteblog_users_with_status_updates"
... WHERE "username" = 'iteblog';
username | email          | encrypted_password
-----+-----+-----
iteblog | iteblog@iteblog.com | 0x877e8c36efa827dbd4cafbc92dd90d76
(1 rows)
```

这样无论表中有多少行 username 为 iteblog 的数据，最终都会显示一行数据。

虽然加了 DISTINCT 关键字，但是 Cassandra 并不是将 username 为 iteblog 的数据全部拿出来，然后再去重的，因为静态列本来在底层就存储了一份，所以不需要再去重。