# 使用一致性级别最大化 DocumentDB 中的可用性和性能

开发人员通常难以在“非常”和“最终”这两种极端的一致性级别之间做出抉择。但事实上，在这两种极端选项之间还有多种一致性级别可供选择。在大部分的真实案例中，应用程序都因为能够在一致性、可用性与延迟之间实现细化的平衡而获得不少好处。DocumentDB 提供四种定义完善的一致性级别以及相关联的性能级别。因此，应用程序开发人员得以在预测范围内针对一致性、可用性和延迟实现平衡。

所有系统资源（包括数据库帐户、数据库、集合、用户和权限）在读取和查询方面始终具有极大程度的一致性。一致性级别仅适用于用户定义的资源。针对用户定义的资源（包括文档、附件、存储过程、触发器和 UDF）所进行的查询和读取操作，DocumentDB 提供四种不同的一致性级别：

* 非常一致性
* 受限停滞一致性
* 会话一致性
* 最终一致性

通过这些细化的定义完善的一致性级别，你可以在一致性、可用性和性能之间实现合理的平衡。这些一致性级别受到可预测的性能级别的支持，可确保应用程序有一致的结果。

## 数据库的一致性级别

你可以配置数据库帐户的默认一致性级别，以应用于数据库帐户下的所有集合（跨所有数据库）。所有对用户定义的资源发出的读取和查询，默认都会使用数据库帐户上指定的默认一致性级别。不过，你可以指定 [x-ms-consistency-level] 请求标头，来降低特定读取/查询请求的一致性级别。DocumentDB 的复制协议支持四种类型的一致性级别，下面有这些级别的简短说明。

[AZURE.NOTE] 之后的版本预计会支持根据每个集合覆盖默认的一致性级别。

**非常**：非常一致性保证写入只有在副本的多数仲裁一直提交的情况下才可见。写入要么由主要副本和辅助副本仲裁一直同步提交，要么被中止。读取始终由多数读取仲裁确认；客户端绝不会看到未提交或不完整的写入，而且始终保证读取最新确认的写入。

非常一致性绝对保证数据一致性，但其读取和写入性能最差。

**受限停滞**：受限停滞一致性保证写入传播的总体顺序，但是读取最多可能会比写入滞后 K 个前缀。读取始终由副本的多数仲裁确认。读取请求的响应可指定其相对新鲜度（以 K 表示）。你可以通过受限停滞设置可配置的停滞阈值（以前缀数或时间表示），以供读取在稳定状态下平衡延迟性与一致性。

受限停滞为读取一致性提供更加容易预测的行为，同时提供最低的延迟写入。因为读取由多数仲裁确认，所以系统提供的读取延迟不是最低的。受限停滞适用于需要非常一致性但非常一致性不可行的情况。如果你将受限停滞一致性的“停滞间隔”配置为任一较大的值，它仍将保留总全局写入顺序。这可以提供比“会话”或“最终”更强的保证。

[AZURE.NOTE] 受限停滞保证只对显式读取请求进行单调读取。写入请求的回显服务器响应不提供受限停滞保证。

**会话**：与非常和受限停滞一致性级别所提供的全局一致性模型不同，“会话”一致性是专为特定客户端会话设计的。会话一致性通常就已足够，因为它提供有保证的单调读取和写入，并且能够读取你自己的写入。会话一致性的读取请求的发出对象是可以提供客户端所请求版本（会话 Cookie 的一部分）的副本。

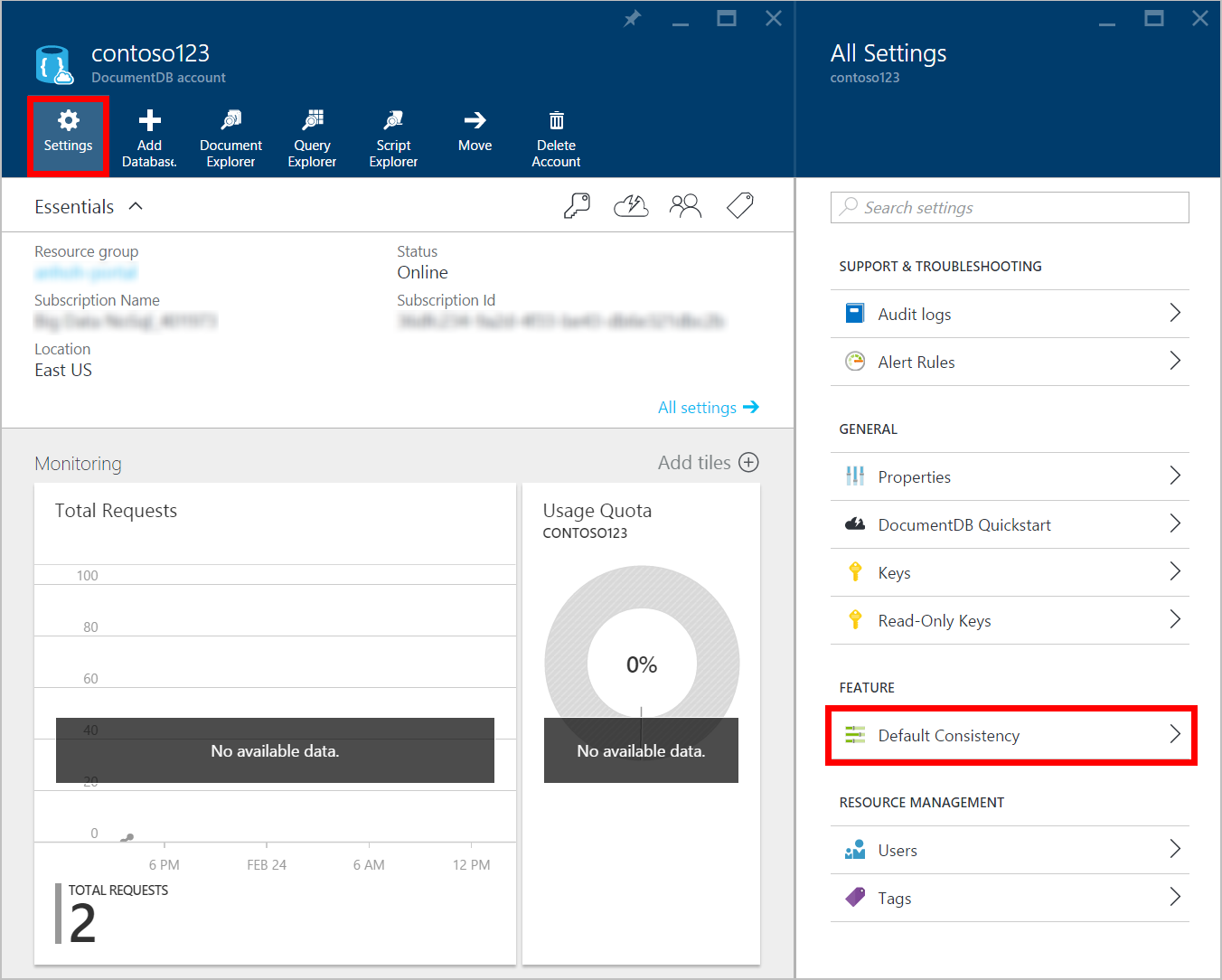
会话一致性为会话提供可预测的读取数据一致性，同时提供最低的延迟写入。在极少数的情况下，读取延迟也很低（这是例外），此时读取会由单个副本提供。

**最终**：最终一致性是最差的一致性形式，在经过一段时间之后，客户端可能会获取比之前看到的值还要旧的值。之后如果没有再收到任何写入，组内的副本最后会只剩一个。读取请求由任意辅助索引提供。

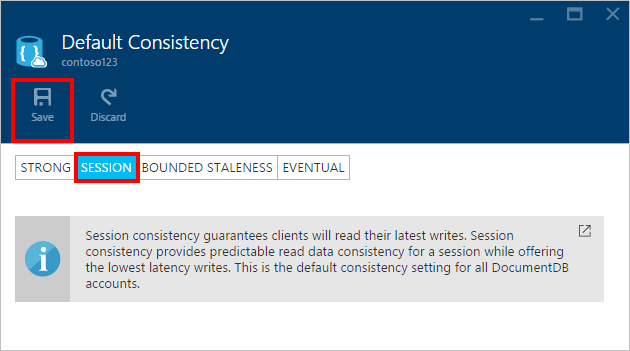
最终一致性提供最差的读取一致性，但是读取和写入的延迟均为最低。

### 更改数据库一致性级别

1. 在 [Azure 门户](https://portal.azure.cn/)的跳转栏中，单击“DocumentDB 帐户”。
2. 在“DocumentDB 帐户”边栏选项卡中，选择要修改的数据库帐户。
3. 在“帐户”边栏选项卡中，如果“设置”边栏选项卡尚未打开，请单击最上方命令栏中的“设置”图标。
4. 在“所有设置”边栏选项卡中，单击“功能”下的“默认一致性”条目。

* 
* 屏幕截图：突出显示“设置”图标和默认一致性条目

1. 在“默认一致性”边栏选项卡中，选择新的一致性级别，然后单击“保存”。

* 
* 屏幕截图：突出显示一致性级别和“保存”按钮的

## 查询的一致性级别

默认情况下，用户定义资源的查询一致性级别与读取相同。默认情况下，每次在集合中插入、替换或删除文档时同步更新索引。这个行为让查询能够使用与文档读取相同的一致性级别。虽然 DocumentDB 针对写入进行了优化，且支持文档持续写入，以及同步索引维护和提供一致的查询服务，但你也可以配置某些集合，使其索引延迟更新。延迟索引编制可大大提高写入性能，非常适合工作负荷主要具有大量读取操作的批量引入方案。

|  |  |  |
| --- | --- | --- |
| 索引模式 | 读取 | 查询 |
| 一致（默认值） | 有非常、受限停滞、会话或最终可供选择 | 有非常、受限停滞、会话或最终可供选择 |
| 延迟 | 有非常、受限停滞、会话或最终可供选择 | 最终 |

与读取请求一样，你可以指定 [x-ms-consistency-level] 请求标头，来降低特定查询请求的一致性级别。

## 后续步骤

如果你想详细了解一致性级别和平衡方案，建议参阅下列资源：

* Doug Terry。Replicated Data Consistency explained through baseball（通过 Baseball 解释的重复数据一致性）。  
  <http://research.microsoft.com/pubs/157411/ConsistencyAndBaseballReport.pdf>
* Doug Terry。Session Guarantees for Weakly Consistent Replicated Data（弱一致性重复数据的会话保证）。  
  <http://dl.acm.org/citation.cfm?id=383631>
* Daniel Abadi。Consistency Tradeoffs in Modern Distributed Database Systems Design: CAP is only part of the story”（现代分布式数据库系统设计中的一致性平衡方案：CAP 只是冰山一角）。  
  <http://computer.org/csdl/mags/co/2012/02/mco2012020037-abs.html>
* Peter Bailis、Shivaram Venkataraman、Michael J. Franklin、Joseph M. Hellerstein、Ion Stoica。Probabilistic Bounded Staleness (PBS) for Practical Partial Quorums（实用部分仲裁的概率性有限过期性 (PBS)）  
  <http://vldb.org/pvldb/vol5/p776_peterbailis_vldb2012.pdf>
* Werner Vogels。Eventual Consistent - Revisited（最终一致 - 重新访问）。  
  <http://allthingsdistributed.com/2008/12/eventually_consistent.html>