由于通过索引查找数据的SQL不需要进行全表扫描，所以，索引能大大提高数据库的性能。

索引是如何工作的？

当你创建一个索引，CockroachDB对你指定的列进行索引，它创建了列的备份，然后对它们的值进行排序（而不对table里的值进行排序）。

列索引建立好后，SQL可以便捷得使用索引来过滤值，而不需要对表进行一行行扫描。在一张大表中，这极大减少了SQL语句执行需要用到的行，执行查询的速度会指数级增长。

例如，当我们在一个INT列建立索引，SQL通过WHERE <indexed column> = 10子句进行过滤，就可以使用索引查找从10开始但小于11的值。相反，如果没有索引，SQL必须评估每个列中的行等于10。

创建

每个表自动会创建一个主键索引，它对主键进行索引，如果没有主键，则为每个行唯一的rowid进行索引。我们建议创建主键，而不要使用rowid，因为创建的主键将提供更好的性能。

primary 索引帮助过滤表的主键，但对于查找其他列没有帮助。不过，你可以使用二级索引来提高查询的性能，你可以如此创建它们：

* 在CREATE TABLE建表时的INDEX子句中使用。除了明确定义的索引之外，CockroachDB还会自动为具有Unique约束的列创建二级索引。
* 对于已存在的表，使用CREATE INDEX
* 使用ALTER TABLE添加UNIQUE约束，它会为约束列自动创建索引

如何创建更有用的二级索引，你可以查看[best practices](http://doc.cockroachchina.baidu.com/#develop/performance-optimization/indexes/#best-practices).

选择

因为每个查询只能使用一个索引，所以CockroachDB选择它计算时需要扫描最少行的索引（即最快的行）。 更多细节查看 [Index Selection in CockroachDB](https://www.cockroachlabs.com/blog/index-selection-cockroachdb-2/).

你还可以强制查询使用特定索引（也称为“index hinting”）来覆盖CockroachDB选择的索引。

存储

CockroachDB将索引直接存储在键值存储中。更多信息查阅：[Mapping Table Data to Key-Value Storage](https://www.cockroachlabs.com/blog/sql-in-cockroachdb-mapping-table-data-to-key-value-storage/).

加锁

CockroachDB创建索引时不会加锁，原因查阅：[schema change procedure](https://www.cockroachlabs.com/blog/how-online-schema-changes-are-possible-in-cockroachdb/).

性能

索引是一种权衡的选择：它们极大提高查询速度，但稍微降低写入速度（因为必须复制和对新值排序）。你创建的第一个索引影响最大，不过其他索引只会产生边际开销。

为了让索引来更好得提高性能，建议可以查看[最佳实践](http://doc.cockroachchina.baidu.com/#develop/performance-optimization/indexes/#best-practices).

最佳实践

我们建议针对所有常用查询建立索引。通过对每个查询的 WHERE and FROM子句来设计最有用的索引，你可以创建索引：

* 索引WHERE子句中所有的列
* 存储仅在FROM子句中的列

索引列

在设计索引时，考虑清楚你需要索引哪些列和列出它们的顺序是很重要的。 以下指引可帮助你做出最佳选择：

* 每个表的主键（我们建议要设主键）会自动创建索引。它创建的索引（称为primary）无法更改，也无法在创建表后更改其主键，因此在建表时，确定如何创建主键索引是非常重要的。
* 索引对查询很有帮助，即使只过滤其列的前缀。例如，如果创建列的索引(A, B, C)，那么过滤(A)或者(A,B)的查询都可以使用索引。但是，如果不过滤(A)的查询将不会从索引中收益。

此功能还允许你避免使用单列索引， 而是使用列作为多列索引中的第一列，这对更多查询很有用。

* 在WHERE子句中使用相等运算符（=或IN）过滤的列应该放在索引前面，在使用不等运算符（<，>）引用的列之前。
* 相同的列，使用不同的排列顺序的索引，在查询中也会产生不同的结果。 更多信息请参阅我们关于[索引选择](https://www.cockroachlabs.com/blog/index-selection-cockroachdb-2/)的博客文章 - 特别是“限制搜索空间”部分。

存储列

存储列可以优化检索其值的查询的性能（即在FROM子句中）但不会过滤它们。 这是因为索引值仅在它们被过滤时才有用，但SQL仍然可以更快地检索它已经扫描的索引中的值，而不是返回到表本身。

但是，要使SQL使用存储列，查询必须过滤同一索引中的另一列。

示例

如果你想优化以下查询的性能：

> **SELECT** col1 **FROM** tbl **WHERE** col1 = 10;

> **SELECT** col1, col2, col3 **FROM** tbl **WHERE** col1 = 10 **AND** col2 > 1;

您可以创建一个存储col3的col1和col2的索引：

> **CREATE** **INDEX** **ON** tbl (col1, col2) STORING (col3);

See Also

* [Inverted Indexes](http://doc.cockroachchina.baidu.com/#performance-optimization/inverted-indexes/)
* [CREATE INDEX](http://doc.cockroachchina.baidu.com/#develop/sql-statements/CREATE-INDEX/)
* [DROP INDEX](http://doc.cockroachchina.baidu.com/#develop/sql-statements/DROP-INDEX/)
* [RENAME INDEX](http://doc.cockroachchina.baidu.com/#develop/sql-statements/RENAME-INDEX/)
* [SHOW INDEX](http://doc.cockroachchina.baidu.com/#develop/sql-statements/SHOW-INDEX/)
* [SQL Statements](http://doc.cockroachchina.baidu.com/#develop/sql-statements/overview/)