# 背景

参考：

<https://disksing.com/understanding-f1-schema-change/>

F1 是 Google 开发的分布式关系数据库，主要服务于 Google 的广告系统，它提供强一致性、高可用性，并支持传统 SQL 查询，近来也常常被称之为所谓的 NewSQL。

F1 是构建于 Spanner 之上的。Spanner 是 Google 开发的全球级数据存储引擎，它保证了数据存储的一致性和可用性，还通过 2PC（两阶段提交）提供了分布式事务读写。在分析 F1 时，我们可以简单地认为 Spanner 是一个全球分布的 kv 数据库。

F1 系统运行时由多台独立的 F1 服务器组成，为了保证整个系统的高可用性，F1 服务器被设计为无状态的，而且不存储数据——节点可以随时上线下线，客户端可以连接至任意节点发送请求。F1 服务器主要职能是将 RDBMS 中的结构化数据映射为可存储于 Spanner 的 kv 对，同时将客户端的 SQL 请求翻译成 get, set, del 等简单的 kv 操作。

Schema 也就是关系数据库中表、列、索引、约束等定义，对应于 SQL 中的 DDL。很显然，Schema 决定了 F1 服务器的具体工作方式，客户端请求的解析和验证由 Schema 决定，之后如何翻译成 kv 操作也由 Schema 决定，Schema 可以被认为是 F1 服务器运行时所依赖的元信息。实践中，F1 服务器运行时自身会缓存一份 Schema 并有一定的机制保持定时更新。

F1 中的 Schema 变更是在线的、异步的，Schema 变更的过程中所有数据保持可用，保持数据一致性，并最大限度的减小对性能的影响。最大的难点在于所有 F1 服务器的 Schema 变更是无法同步的，也就是说不同的 F1 服务器会在不同的时间点切换至新 Schema。

说明：在TDSQL2.0、TDSQL2.5中都采用了proxy/SQLEngine之间广播的方式同步，这种效率不是很高，并不是真正的Online-DDL，TDSQL3.0实现了真正的F1 Online-DDL。

由于所有的 F1 服务器共享同一个 kv 存储引擎，Schema 的异步更新可能造成严重的数据错乱。例如我们发起给一次添加索引的变更，更新后的节点会很负责地在添加一行数据的同时写入一条索引，随后另一个还没来得及更新的节点收到了删除同一行数据的请求，这个节点还完全不知道索引的存在，自然也不会去删除索引了，于是错误的索引就被遗留在数据库中。

# 原理

参考：

[《Spanner: Google’s Globally-Distributed Database》论文翻译](http://blog.mrcroxx.com/posts/paper-reading/spanner-osdi2012/)

[谷歌 F1 Online DDL的关键点：状态间兼容性](https://zhuanlan.zhihu.com/p/120719499)

[Google F1是如何做Schema变更的](https://juejin.cn/post/6844903623109902350)

[分布式 Schema 变更在 Google F1 的实践](https://zhuanlan.zhihu.com/p/367041904)

## 租约

F1中 Schema 以特殊的 kv 对存储于 Spanner 中，同时每个 F1 服务器在运行过程中自身也维护一份拷贝。为了保证同一时刻最多只有 2 份 Schema 生效，F1 约定了长度为数分钟的 Schema 租约，所有 F1 服务器在租约到期后都要重新加载 Schema。如果节点无法重新完成续租，它将会自动终止服务并等待被集群管理设施重启。

## 中间状态

前面已经提过，F1在Schema变更的过程中，会把一次Schema的变更拆解为多个逐步递进的中间状态。实际上我们并不需要针对每种Schema变更单独设计中间状态，总共只需要两种就够了：delete-only和write-only。

delete-only指的是Schema元素的存在性只对删除操作可见。

例如当某索引处于delete-only状态时，F1服务器中执行对应表的删除一行数据操作时能“看到”该索引，所以会同时删除该行对应的索引，与之相对的，如果是插入一行数据则“看不到”该索引，所以F1不会尝试新增该行对应的索引。

具体的，如果Schema元素是table或column，该Schema元素只对delete 语句生效；如果Schema元素是index ，则只对delete和update语句生效，其中update语句修改index的过程可以认为是先delete后再insert，在delete-only 状态时只处理其中的 delete 而忽略掉 insert 。

总之，只要某Schema元素处于delete-only状态，F1保证该Schema元素对应的kv对总是能够被正确地删除，并且不会为此Schema元素创建任何新的kv对。

write-only指的是Schema元素对写操作可见，对读操作不可见。

例如当某索引处于write-only状态时，不论是insert、delete，或是update，F1都保证正确的更新索引，只是对于查询来说该索引仍是不存在的。

简单的归纳下就是write-only状态的Schema元素可写不可读。