**Follow-the-Workload**，指的是CockroachDB能够根据请求负载源的网络位置情况，动态调整CockroachDB内部的运行机制，优化读延迟。

本小节将简单地介绍该功能特性。

概要

基本概念

| **英文** | **中文** | **简述** |
| --- | --- | --- |
| **Range** | - | CockroachDB存储所有用户数据和系统数据，整体逻辑上是一个有序的key-value集合，物理上根据key划分成多个区间，每个区间是一个Range。 |
| **Range Replica** | Range副本冗余 | CockroachDB对每个Range做数据冗余（默认情况下是3份副本），并分别存储在不同的节点上。 |
| **Range Lease** | Range租约 | 一个Range的所有副本当中只有一个副本持有Range租约，持有者接受和协调关于该Range的所有读写请求。 |

工作原理

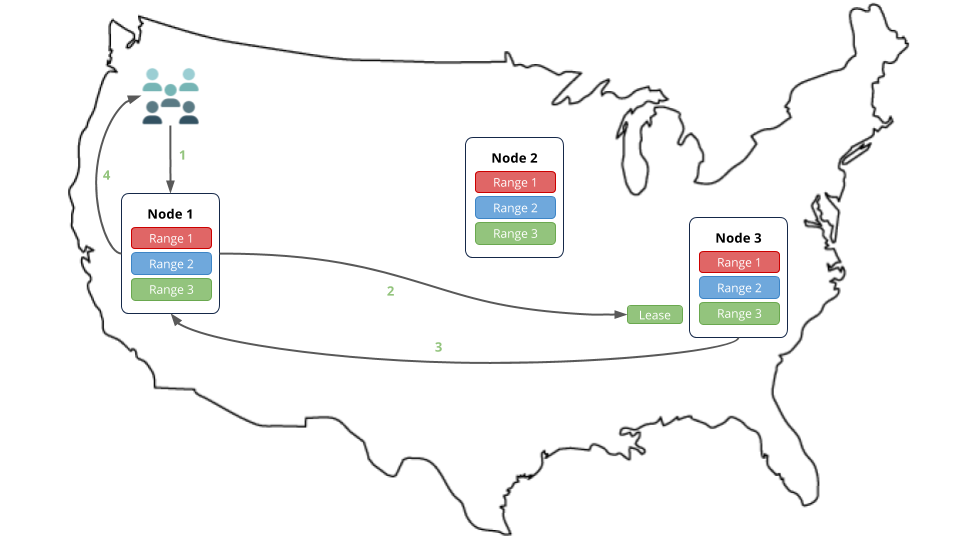
**Follow-the-workload**处理读请求是基于Range租约。绝大部分的读请求能够直接访问到拥有Range租约的Range副本所在的节点并进行处理，避免了请求再路由到Range租约持有者所在节点的网络代价。值得注意的是，具有Range租约的节点同时接受处理写请求，能够保证该Range数据版本是最新的。

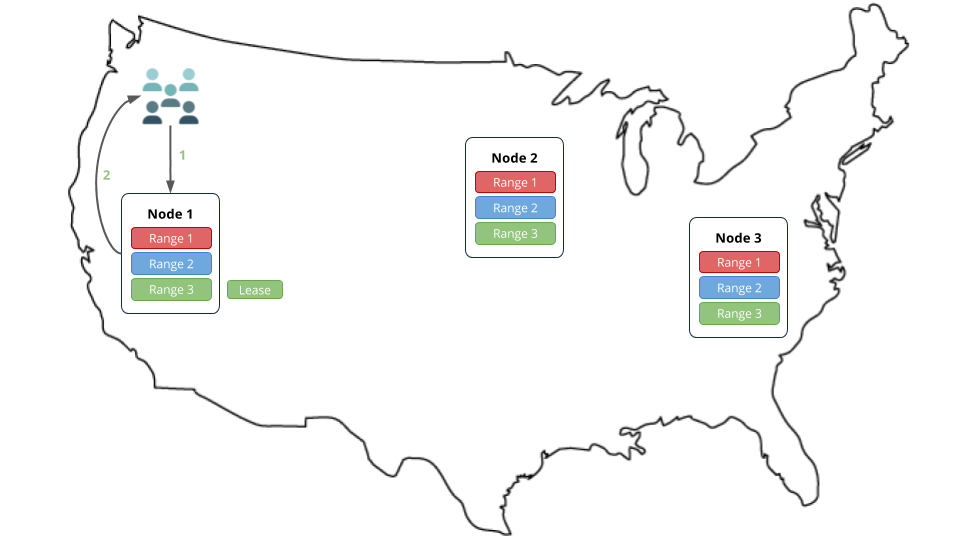
这种方式能够保证读请求的处理速度。假设读请求“随机地”发送到持有该Range的节点上，然而此时不能保证该节点是否持有该Range的Range租约，大概率需要通过该节点将请求转发给Range租约持有者的节点去处理，而这个转发过程受网络延迟的影响。

为解决这个问题，我们首先需要利用--localityFlag将集群CockroachDB节点按物理划分成多个拓扑节点，每个拓扑节点内CockroachDB节点之间的网络传输代价较小，而拓扑节点之间CockroachDB节点的网络传输代价较大。此时CockroachDB就能够实时地根据该Range在不同拓扑节点上的读负载情况，将Range租约优先分配给负载更大的拓扑节点中的CockroachDB节点。

**TIPS:** 为开启**Follow-the-workload**功能，用户需要在集群中每个CockroachDB节点启动时手动设置--localityFlag。

案例

如下案例中，很多客户端向位于美国西部的节点1发出针对Range3数据的读请求，而此时持有Range租约的Range3副本在美国东部的节点3，此时节点1将把读请求路由到节点3去处理。 

而通过--localityFlag实现的**Follow-the-Workload**功能，能够在分析出读请求负载集中在美国西部的节点，则Range租约优先给予美国西部的节点1，此时能够大大地避免请求在美国东西部之间的传输代价。 

教程

Step 1: 安装环境

包括：

* [CockroachDB](http://doc.cockroachchina.baidu.com/#quick-start/install-cockorachdb/)
* [Go](https://golang.org/doc/install)运行环境，版本不低于1.9
* [comcast](https://github.com/tylertreat/comcast)网络模拟工具

go get github.com/tylertreat/comcast

* [kv](https://github.com/cockroachdb/loadgen/tree/master/kv)生成器

**go** get github.**com**/cockroachdb/loadgen/kv

Step 2: 分析网络延迟情况

**Follow-the-Workload**功能针对的是集群中节点之间有较高网络延迟的情况。此处我们使用comcast工具模拟节点之间的高网络延迟：

comcast --device lo0 --latency 100

**NOTE:** 对于--device的参数值，Mac机器上为lo0，linux机器上为lo，其他机器请通过ifconfig命令查看127.0.0.1对应的网络接口情况。

上述命令人为地使得所有经过局域网的请求新增100ms的延迟。

Step 3: 启动节点

启动美国西部节点

cockroach start --insecure --locality=region=us-west --host=localhost --store=follow1 --port=26257 --http-port=8080 --join=localhost:26257,localhost:26258,localhost:26259

启动美国中西部节点

cockroach start --insecure --locality=region=us-midwest --host=localhost --store=follow2 --port=26258 --http-port=8081 --join=localhost:26257,localhost:26258,localhost:26259

启动美国东部节点

cockroach start --insecure --locality=region=us-east --host=localhost --store=follow3 --port=26259 --http-port=8082 --join=localhost:26257,localhost:26258,localhost:26259

Step 4: 初始化集群

cockroach init --insecure --host=localhost --port=26257

Step 5: 模拟美国东部的负载

利用kv工具生成请求，指向美国东部的节点：

kv -duration 1m -concurrency 32 -read-percent 100 -max-rate 100 'postgresql://root@localhost:26259?sslmode=disable'

该命令生成了32个客户端连接，每个连接持续1分钟发出读请求到指定节点，总体上保持100OPS的速度。

\_elapsed\_\_\_errors\_\_ops/**sec**(inst)\_\_\_ops/**sec**(cum)\_\_p95(ms)\_\_p99(ms)\_pMax(ms)

1s 0 23.0 23.0 838.9 838.9 838.9

2s 0 111.0 66.9 805.3 838.9 838.9

3s 0 100.0 78.0 209.7 209.7 209.7

4s 0 99.9 83.5 209.7 209.7 209.7

5s 0 100.0 86.8 209.7 209.7 209.7

...

**NOTE:** 可以看到kv工具的请求有大概200ms左右的延迟，原因是因为comcast在本地网络的延迟100ms设置生效。对comcast配置更高级的网络规则可以避免这种情况发生。

等待kv工具执行完毕。

Step 6: 确认Range租约持有节点的位置

查看集群各节点id情况

cockroach node status --insecure --port=26257

+----+-----------------+--------+---------------------+---------------------+---------+

| id | address | build | updated\_at | started\_at | is\_live |

+----+-----------------+--------+---------------------+---------------------+---------+

| 1 | localhost:26257 | v2.0.3 | 2018-07-05 07:58:17 | 2018-07-05 07:53:57 | true |

| 2 | localhost:26258 | v2.0.3 | 2018-07-05 07:58:19 | 2018-07-05 07:53:58 | true |

| 3 | localhost:26259 | v2.0.3 | 2018-07-05 07:58:19 | 2018-07-05 07:53:59 | true |

+----+-----------------+--------+---------------------+---------------------+---------+

(3 rows)

根据ip地址可以看见美国东部的节点对应id为3。

开启交互式shell

cockroach sql --insecure --port=26257

查看Range租约的持有者

SHOW EXPERIMENTAL\_RANGES FROM TABLE test.kv;

+-----------+---------+----------+----------+--------------+

| Start Key | End Key | Range ID | Replicas | Lease Holder |

+-----------+---------+----------+----------+--------------+

| NULL | NULL | 29 | {1,2,3} | 3 |

+-----------+---------+----------+----------+--------------+

(1 row)

可以看到Range副本对应的Range租约持有者，正是Step 5当中利用kv工具针对性地生成读负载的美国东部节点（Lease Holder = 美东节点id = 3），**Follow-the-Workload**功能生效。

Step 7: 模拟美国西部的负载

利用kv工具生成请求，指向美国西部的节点：

kv -duration 7m -concurrency 32 -read-percent 100 -max-rate 100 'postgresql://root@localhost:26257?sslmode=disable'

该命令生成了32个客户端连接，每个连接持续7分钟发出读请求到指定节点，总体上保持100OPS的速度。 等待kv工具执行完毕。

Step 8: 确认Range租约持有节点的位置

查看集群各节点id情况

cockroach node status --insecure --port=26257

+----+-----------------+--------+---------------------+---------------------+---------+

| id | address | build | updated\_at | started\_at | is\_live |

+----+-----------------+--------+---------------------+---------------------+---------+

| 1 | localhost:26257 | v2.0.3 | 2018-07-05 07:58:17 | 2018-07-05 07:53:57 | true |

| 2 | localhost:26258 | v2.0.3 | 2018-07-05 07:58:19 | 2018-07-05 07:53:58 | true |

| 3 | localhost:26259 | v2.0.3 | 2018-07-05 07:58:19 | 2018-07-05 07:53:59 | true |

+----+-----------------+--------+---------------------+---------------------+---------+

(3 rows)

根据ip地址可以看见美国西部的节点对应id为1。

开启交互式shell

cockroach sql --insecure --port=26257

查看Range租约的持有者

SHOW EXPERIMENTAL\_RANGES FROM TABLE test.kv;

+-----------+---------+----------+----------+--------------+

| Start Key | End Key | Range ID | Replicas | Lease Holder |

+-----------+---------+----------+----------+--------------+

| NULL | NULL | 29 | {1,2,3} | 1 |

+-----------+---------+----------+----------+--------------+

(1 row)

可以看到Range副本对应的Range租约持有者，真是Step 7当中利用kv工具针对性地生成读负载的美国西部节点（Lease Holder = 美西节点id = 1），**Follow-the-Workload**功能生效。

Step 9: 关闭集群

在各个终端使用Ctrl+C终止程序。 如不再需要测试用的集群，可使用如下指令移除集群：

**rm** -rf follow1 follow2 follow3

**NOTE:** 对于最后两个节点，可能需要花大约1min左右的时间去终止程序。原因是最后只剩下1个节点在集群当中的时候，由于已经丧失了集群当中冗余副本的绝大多数（此情况下是2/3），集群不再允许操作。如需加速终止程序的进程，可以尝试多次Ctrl+C操作。

Step 10: 停止模拟网络延迟

执行以下命令：

*comcast* --*device* *lo0* --*stop*