

跨数据源(catalog)的一些研究

由于组内项目面临大面积重构，初步的方案决定按业务重新划分数据源，将他们互相解耦出来；但由于有些页面报表，离线数据等需求是需要跨源去合并数据的，暂时的数据源只有Mysql，但不排除以后会使用ES、HIVE等其他源，所以在市场上开始找一些可以使用的高性能跨源组件。

通过查阅一些资料以及跟一些有类似经验的同学聊了之后是有了一些方案的：

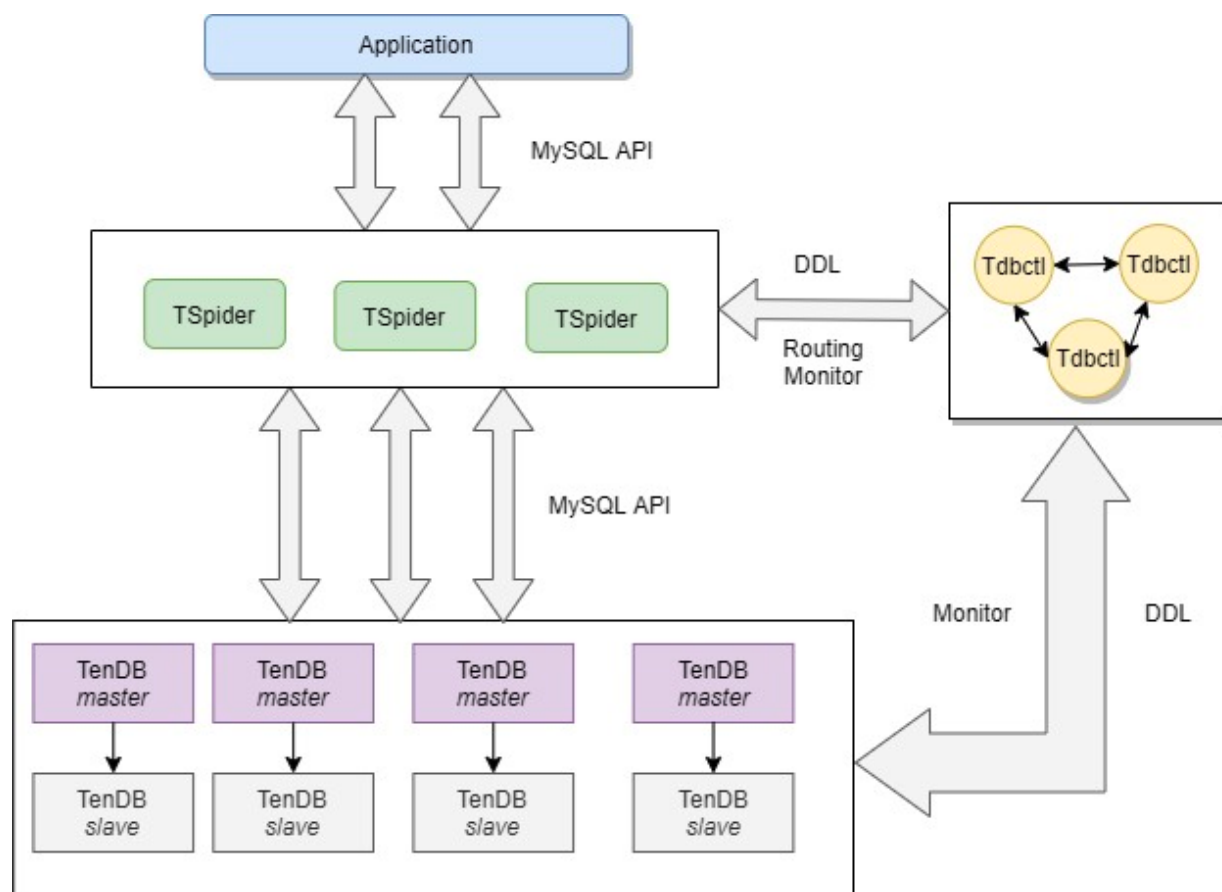
- 按需分库查出数据，然后在内存中用代码处理返回（内存占用大，可能导致频繁GC；并且性能无法得到保证，单机CPU性能有限；业务处理复杂，如果多表join，代码将变得难以维护）
- 使用Mysql的FEDERATED引擎来建立远程连接表（类似于Oracle的DBlink,但是只能跨Mysql数据源，并且性能也不是很高，对于远程表需要自己维护）
- 公司内部一些比较热的组件：Tspider（开源）、SuperSql（闭源）、PrestoDB（开源）前两者都是腾讯的组件，后面那个是Facebook开源的组件（还有个PrestoSql就是以前PrestoDB的团队单干的一个相似组件）

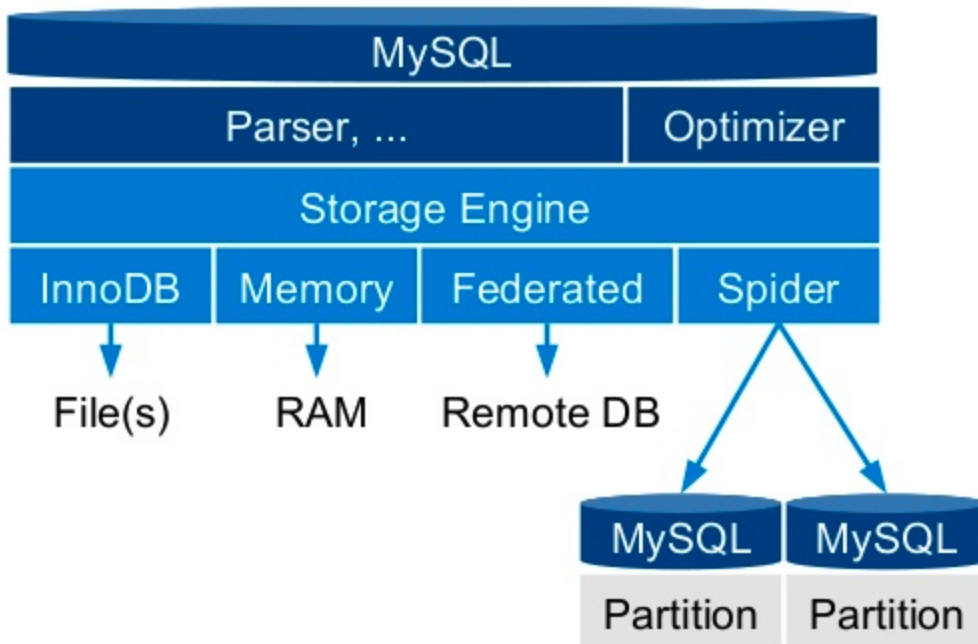
通过这些方面的对比，上述的那些组件是比较适合这次的技术选型；那么我们一起来看来上述三个组件的一些不同和优势：

- Tspider是分布式Mysql，透明分库分表、高可用的MySQL集群服务，透明及在线的扩容及缩容；Tspider也支持范围查询或跨分区查询及更新，但效率会差些，跨表查询的效率最差；所以它适用于点查询
- Presto使用场景主要是adhoc(即席查询：尽可能快的执行自定义SQL)，适合做探索性分析，跨数据源的功能不是很常用，但它支持了非常多的数据源，性能也有一定的保证；
- SuperSql尚未开源，就先看下[这篇文章](#)吧，有详细的介绍和性能测试；

Tspider

我们看一下来自Tspider官网的[整体架构](#)（这里得吐槽一下，得F12才能拿到page url）：





上面官方文档介绍的很清楚，基本看一遍就能懂；

简单来讲就是TSpider由三大组件组成

- Tspider (TenDB Cluster集群的接入层)
- TenDB (TenDB Cluster的数据存储层)
- Tdbctl (TenDB Cluster集群的中央控制模块)

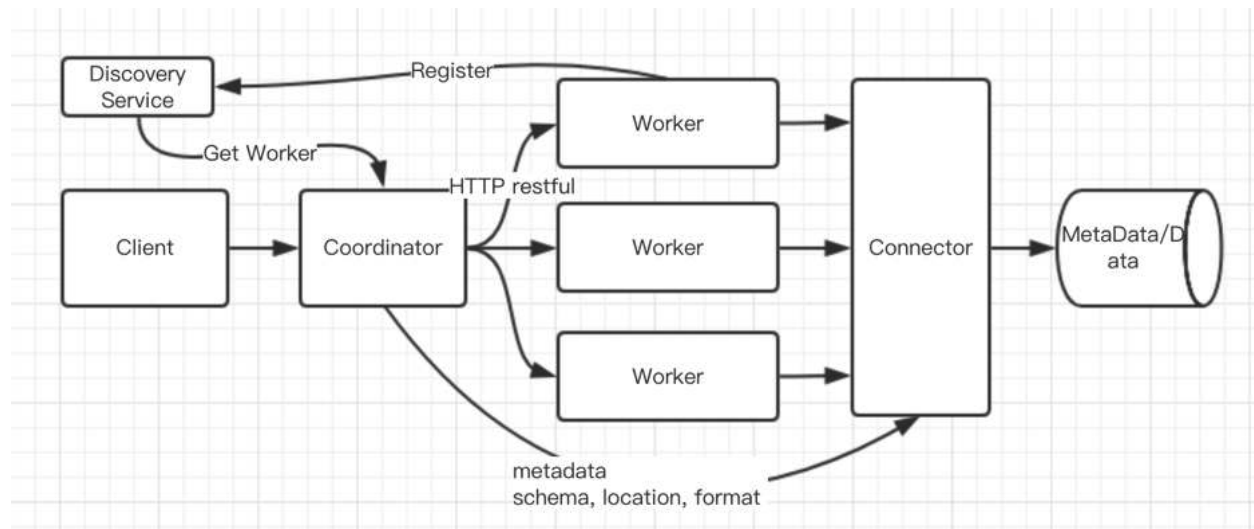
它参考了Mysql的分区表而做了一个Spider引擎，是基于开源框架Spider3.1改造升级的；但这个引擎不像分区表一样，它是不做数据存储的，是无状态的，它会根据分区配置将数据发送到TenDB上存储，如果是查询请求他也将去对应分区的TenDB上进行查询；那么这里可能就会产生一些问题，如果我们需要查询多个跨Partition的数据，那么最后的聚合效率肯定是要大打折扣的，就像我们分区表查询多个分区数据一样，引擎需要对多个分区进行查询，并且聚合返回，所以这也是它性能瓶颈的地方。看了下社区的发展路线，上面列举的点还是对SQL性能的优化，以及一些功能的扩展（贴近于Mysql的，比如全局唯一递增ID等）；

- 使用性方面，由于Tspider是基于Mysql开发的，所以他是后端语言无障碍的，可以使用Mysql的就可以
- 安全性方面，可以规避安全问题，全部交给Mysql来做即可

- 社区算不上活跃，但是公司内部使用较多，也可以得到更多实战的经验和教训；
- 业务支持方面，暂时只支持Mysql，其他数据源无法得到支持
- 性能方面，毕竟是个分布式Mysql组件，点查询给力，但是对于大数据量多分区join效率不高

Presto

前面介绍了下Presto用于即席查询较多，先看看它的Use Cases，然后可以来看看他的流程架构：



它的整个流程其实在上图已经画的很明确，其中Connector可以有多个也就是我们常说的Catalog，比如我们使用的Mysql、Hive等，它是一个基于内存的分布式查询引擎；

- 适用性方面，其中的Client可以直接使用Http请求代替，虽然官方提供了go、python等开源客户端；但是最简单的restful 的http请求也是可以支持的，这点还是比较好的，减少了对于客户端的依赖。
- 安全性上面，它提供了诸多的安全认证，最常用的我们可以在Presto Coordinator上对客户端做配置认证
- 社区方面PrestoDB在github上面是少有的star有11k的java项目，社区比较活跃
- 业务支持方面，Presto提供丰富的数据源的支持，并且对于数据源的性能也比传统的MPP框架有更高的性能，唯一美中不足的地方是，他只支持标准SQL，那么Mysql自

有的一些语法可能需要改造才能支持，比如说limit、group by等，并且他对于跨源join的数据格式要求是一致的，否则需要进行cast

- 性能方面，除了官网的性能测试，我也自己进行了一些业务场景的测试（测试数据已脱敏），大致是秒级返回，对于并发数跟QPS大致是成线程成长趋势。

我前面介绍了[怎么搭建Presto](#)，贴一下这次测试的配置

Coordinator配置

etc/node.properties

```
node.environment=production
node.id=ffffffff-ffff-ffff-ffff-ffffffffffff1
node.data-dir=xxx/data
```

etc/config.properties

```
coordinator=true
node-scheduler.include-coordinator=true#配置为既是Coordinator也是Worker
http-server.http.port=8080
query.max-memory=8GB
query.max-memory-per-node=2GB
query.max-total-memory-per-node=4GB
discovery-server.enabled=true
discovery.uri=http://xxx:8080
```

etc/jvm.config

```
-server
-Xmx6G
-XX:+UseG1GC
-XX:G1HeapRegionSize=32M
-XX:+UseGCOverheadLimit
-XX:+ExplicitGCInvokesConcurrent
-XX:+HeapDumpOnOutOfMemoryError
-XX:+ExitOnOutOfMemoryError
```

Worker配置

etc/node.properties

```
node.environment=production
node.id=ffffffff-ffff-ffff-ffff-ffffffffffff2
node.data-dir=xxx/data
```

etc/config.properties

```
coordinator=false
http-server.http.port=8080
query.max-memory=8GB
query.max-memory-per-node=1GB
query.max-total-memory-per-node=2GB
discovery.uri=http://xxx:8080
```

etc/jvm.config

```
-server
-Xmx6G
-XX:+UseG1GC
-XX:G1HeapRegionSize=32M
-XX:+UseGCOverheadLimit
-XX:+ExplicitGCInvokesConcurrent
-XX:+HeapDumpOnOutOfMemoryError
-XX:+ExitOnOutOfMemoryError
```

正常情况下coordinator机子的情况


```
≡ presto_test(11.185.148.111) ≡ presto_test(9.146.110.208) (1) ≡ presto_test(11.185.148.111) (1) ≡ presto_test(9.146.110.208) (2) × ≡ script_m

top - 09:23:46 up 12 days, 17:29, 4 users, load average: 0.00, 0.16, 0.22
Tasks: 68 total, 1 running, 67 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 0.4 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 8388608 total, 7926644 free, 88824 used, 373140 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 8037798 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR  S  %CPU  %MEM     TIME+ COMMAND
    1 root        20   0   42052   3456   2300  S   0.0   0.0   0:48.21 systemd
   46 root        20   0   25664   1628   1348  S   0.0   0.0   0:03.47 systemd-logind
   47 root        20   0  379612   9872   9132  S   0.0   0.1   0:29.21 rsyslogd
   51 dbus        20   0   25936   1636   1300  S   0.0   0.0   0:07.14 dbus-daemon
   72 root        20   0   25308    788    600  S   0.0   0.0   0:00.00 atd
  168 root        20   0   11864   2732   1252  S   0.0   0.0   0:33.63 watchdog.sh
 2276 root        20   0   19176 10004   1252  S   0.0   0.1   2:52.08 safe_TsysLXCAge
 2570 root        20   0   13232   4052   1252  S   0.0   0.0   0:44.22 safe_TsysLXCPro
 2600 root        20   0   16572   1604    836  S   0.0   0.0   3:15.38 TsysLXCProxy
 2725 root        20   0   22856   7596   7112  S   0.0   0.1   0:24.74 TsysLXCagent
 3071 root        20   0   38180    560    216  S   0.0   0.0   0:00.00 gseMaster
 3073 root        20   0  2736272 11492   2220  S   0.0   0.1   6:59.60 agentWorker
 7026 root        20   0   82172   3324   2480  S   0.0   0.0   0:00.02 sshd
 7030 root        20   0   82172   3316   2472  S   0.0   0.0   0:00.00 sshd
 8445 root        20   0  134208   4708   3472  S   0.0   0.1   0:00.01 sshd
 8521 root        20   0   10956   2124   1532  S   0.0   0.0   0:00.01 bash
 9821 root        20   0  134208   5056   3820  S   0.0   0.1   0:00.01 sshd
 9824 paulbli    20   0  134208   2000    756  S   0.0   0.0   0:00.21 sshd
 9825 paulbli    20   0   10936   1996   1468  S   0.0   0.0   0:00.00 bash
 9862 root        20   0   52508   1832   1400  S   0.0   0.0   0:00.00 sudo
 9863 root        20   0   10972   2152   1544  S   0.0   0.0   0:00.01 bash
11058 root        20   0  134208   5052   3820  S   0.0   0.1   0:00.00 sshd
11061 paulbli    20   0  134208   1996    756  S   0.0   0.0   0:00.00 sshd
11062 paulbli    20   0   10936   1996   1468  S   0.0   0.0   0:00.00 bash
11086 root        20   0   52508   1832   1400  S   0.0   0.0   0:00.00 sudo
11087 root        20   0   10072   2124   1512  S   0.0   0.0   0:00.00 bash
11726 root        20   0  134208   5052   3820  S   0.0   0.1   0:00.01 sshd
11729 paulbli    20   0  134344   1996    756  S   0.0   0.0   0:00.31 sshd
11730 paulbli    20   0   10936   1996   1468  S   0.0   0.0   0:00.00 bash
11754 root        20   0   52508   1828   1400  S   0.0   0.0   0:00.00 sudo
11755 root        20   0   10972   2116   1504  S   0.0   0.0   0:00.02 bash
14414 root        20   0   37808  20236     8  S   0.0   0.2   0:00.78 secu-tcs-agent
18748 root        20   0  477040   4588   3412  S   0.0   0.1   0:23.56 agent
18766 root        20   0   20080    784    496  S   0.0   0.0   0:00.72 agentPlugInD
18787 root        20   0   23804   2108    900  S   0.0   0.0   0:26.43 base
18805 root        20   0   22272    992    784  S   0.0   0.0   0:01.59 tcvmstat
18838 root        20   0    9976    896    356  S   0.0   0.0   0:01.74 sysddd
19001 nsd        20   0  414840   1240    936  S   0.0   0.0   0:00.01 nsd
19156 root        20   0   64968   1628   1076  S   0.0   0.0   0:00.00 nsdcd
20091 root        20   0   53288   2144  1428  R   0.0   0.0   0:00.13 top
20827 root        20   0    7420    380    292  S   0.0   0.0   0:00.00 sleep
20916 root        20   0    7420    380    292  S   0.0   0.0   0:00.00 sleep
20998 root        20   0    7420    380    292  S   0.0   0.0   0:00.00 sleep
22879 root        20   0  120236   4576   1736  S   0.0   0.1   0:00.06 nginx
27973 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27974 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27975 nginx        20   0  120240   3456    280  S   0.0   0.0   0:00.00 nginx
27976 nginx        20   0  120240   3456    280  S   0.0   0.0   0:00.00 nginx
27977 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27978 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27979 nginx        20   0  120240   3460    284  S   0.0   0.0   0:00.00 nginx
27980 nginx        20   0  120240   3456    280  S   0.0   0.0   0:00.00 nginx
27981 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27982 nginx        20   0  120240   3456    280  S   0.0   0.0   0:00.00 nginx
27983 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27984 nginx        20   0  120252   3508    332  S   0.0   0.0   0:00.00 nginx
27985 nginx        20   0  120240   3456    280  S   0.0   0.0   0:00.00 nginx
27986 nginx        20   0  120252   3500    324  S   0.0   0.0   0:00.00 nginx
27987 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27988 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27989 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27990 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27991 nginx        20   0  120252   3508    332  S   0.0   0.0   0:00.01 nginx
27992 nginx        20   0  120252   3508    332  S   0.0   0.0   0:00.00 nginx
27993 nginx        20   0  120252   3512    336  S   0.0   0.0   0:00.00 nginx
27994 nginx        20   0  120252   3500    324  S   0.0   0.0   0:00.01 nginx
27995 nginx        20   0  120252   3508    332  S   0.0   0.0   0:00.02 nginx
27996 nginx        20   0  120252   3496    320  S   0.0   0.0   0:00.05 nginx
```

启动presto-server之后的coordinator机子的情况


```
≡ presto_test(11.185.148.111)  ≡ presto_test(9.146.110.208) (1)  ≡ presto_test(11.185.148.111) (1) ×  ≡ presto_test(9.146.110.208) (2)

top - 09:25:35 up 12 days, 17:35, 3 users, load average: 0.15, 0.22, 0.35
Tasks: 42 total, 1 running, 41 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.8 us, 0.3 sy, 0.0 ni, 98.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 8388608 total, 7508916 free, 477716 used, 401976 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 7671574 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 91504 root        20   0 28.873g 480676 88124 S   3.7   5.7   0:20.97 presto-server
    1 root        20   0  44716    2740   1332 S   0.0   0.0   0:38.86 systemd
   50 root        20   0 328856   6524   5796 S   0.0   0.1   0:28.12 rsyslogd
   54 dbus        20   0  26044    1236    744 S   0.0   0.0   0:07.07 dbus-daemon
   92 root        20   0  11864    2700   1220 S   0.0   0.0   0:30.91 watchdog.sh
  129 root        20   0  25532    1192    904 S   0.0   0.0   0:04.04 systemd-logind
  133 root        20   0  25308     636    444 S   0.0   0.0   0:00.00 atd
  2478 root        20   0  37864   6092   5812 S   0.0   0.1   0:46.76 systemd-journal
  2802 root        20   0  17596   8412   1196 S   0.0   0.1   2:22.39 safe_TsysLXCAGE
 3097 root        20   0  14848   5564   1200 S   0.0   0.1   0:46.21 safe_TsysLXCPro
 3125 root        20   0  16572   1964   1200 S   0.0   0.0   2:24.72 TsysLXCProxy
 3255 root        20   0  22856   7992   7508 S   0.0   0.1   0:22.11 TsysLXCAGENT
 3626 root        20   0   38180     544    200 S   0.0   0.0   0:00.00 gseMaster
 3628 root        20   0 2736276 11544   2232 S   0.0   0.1   5:13.85 agentWorker
 6130 nscd        20   0 941172   1552    904 S   0.0   0.0   0:44.84 nscd
 7205 root        20   0  21880   1396    792 S   0.0   0.0   0:07.50 crond
 7217 root        20   0   82172   1608    760 S   0.0   0.0   0:00.02 sshd
 7221 root        20   0   82172   1632    784 S   0.0   0.0   0:00.00 sshd
31389 root        20   0 477036   4084   2992 S   0.0   0.0   0:25.01 agent
31400 root        20   0 200080    768    484 S   0.0   0.0   0:00.69 agentPlugInD
31420 root        20   0  23812   2132    880 S   0.0   0.0   0:23.42 base
31433 root        20   0  22272    796    592 S   0.0   0.0   0:01.65 tcvmstat
31469 root        20   0   9976     852    312 S   0.0   0.0   0:00.19 sysddd
50391 polkitd    20   0 526800   4208    884 S   0.0   0.1   0:00.01 polkitd
77730 root        20   0  37812 20244     8 S   0.0   0.2   0:00.98 secu-tcs-agent
81480 root        20   0 134208   2160    928 S   0.0   0.0   0:00.01 sshd
81483 paulbli    20   0 134208   1576    320 S   0.0   0.0   0:00.59 sshd
81484 paulbli    20   0 10936   1780   1248 S   0.0   0.0   0:00.00 bash
81508 root        20   0  52508   1220    792 S   0.0   0.0   0:00.00 sudo
81509 root        20   0  10992   1980   1348 S   0.0   0.0   0:00.01 bash
82316 root        20   0 134208   2160    928 S   0.0   0.0   0:00.00 sshd
82319 paulbli    20   0 134208   1628    368 S   0.0   0.0   0:00.17 sshd
82320 paulbli    20   0 10936   1780   1248 S   0.0   0.0   0:00.00 bash
82344 root        20   0  52508   1220    792 S   0.0   0.0   0:00.00 sudo
82345 root        20   0  10992   1936   1304 S   0.0   0.0   0:00.00 bash
83574 root        20   0 134208   2088    852 S   0.0   0.0   0:00.01 sshd
83659 root        20   0  10936   1860   1284 S   0.0   0.0   0:00.00 bash
83725 root        20   0  64968   2228     0 S   0.0   0.0   0:01.34 nslcd
90957 root        20   0  57392   2140   1444 R   0.0   0.0   0:00.08 top
91466 root        20   0   7420    380    292 S   0.0   0.0   0:00.00 sleep
91484 root        20   0   7420    384    292 S   0.0   0.0   0:00.00 sleep
91794 root        20   0   7420    384    292 S   0.0   0.0   0:00.00 sleep
```

启动presto-server之后worker机子的情况

presto_test(11.185.148.111)										
presto_test(9.146.110.208) (1)										
presto_test(11.185.148.111) (1)										
presto_test(9.146.110.208) (2) ×										
script_m										
top - 09:26:35 up 12 days, 17:32, 4 users, load average: 0.09, 0.14, 0.21										
Tasks: 69 total, 1 running, 68 sleeping, 0 stopped, 0 zombie										
%Cpu(s): 0.7 us, 0.3 sy, 0.0 ni, 99.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st										
KiB Mem : 8388608 total, 7284636 free, 729140 used, 374832 buff/cache										
KiB Swap: 0 total, 0 free, 0 used. 7397482 avail Mem										
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
21535	root	20	0	24.616g	717440	88052	S	2.7	8.6	0:20.78 presto-server
1	root	20	0	42052	3456	2300	S	0.0	0.0	0:48.21 systemd
46	root	20	0	25664	1628	1348	S	0.0	0.0	0:03.47 systemd-logind
47	root	20	0	379612	9872	9132	S	0.0	0.1	0:29.21 rsyslogd
51	dbus	20	0	25936	1636	1300	S	0.0	0.0	0:07.14 dbus-daemon
72	root	20	0	25308	788	600	S	0.0	0.0	0:00.00 atd
168	root	20	0	11864	2732	1252	S	0.0	0.0	0:33.63 watchdog.sh
2276	root	20	0	19112	9964	1252	S	0.0	0.1	2:52.18 safe_TsysLXCAGE
2570	root	20	0	13232	4052	1252	S	0.0	0.0	0:44.24 safe_TsysLXCPro
2600	root	20	0	16572	1604	836	S	0.0	0.0	3:15.42 TsysLXCProxy
2725	root	20	0	22856	7596	7112	S	0.0	0.1	0:24.74 TsysLXCAGENT
3071	root	20	0	38180	560	216	S	0.0	0.0	0:00.00 gseMaster
3073	root	20	0	2736272	11500	2228	S	0.0	0.1	6:59.67 agentWorker
7026	root	20	0	82172	3324	2480	S	0.0	0.0	0:00.02 sshd
7030	root	20	0	82172	3316	2472	S	0.0	0.0	0:00.00 sshd
8445	root	20	0	134208	4708	3472	S	0.0	0.1	0:00.01 sshd
8521	root	20	0	10956	2124	1532	S	0.0	0.0	0:00.01 bash
9821	root	20	0	134208	5056	3820	S	0.0	0.1	0:00.01 sshd
9824	paulbli	20	0	134208	2000	756	S	0.0	0.0	0:00.25 sshd
9825	paulbli	20	0	10936	1996	1468	S	0.0	0.0	0:00.00 bash
9862	root	20	0	52508	1832	1400	S	0.0	0.0	0:00.00 sudo
9863	root	20	0	10972	2152	1544	S	0.0	0.0	0:00.01 bash
11058	root	20	0	134208	5052	3820	S	0.0	0.1	0:00.00 sshd
11061	paulbli	20	0	134208	1996	756	S	0.0	0.0	0:00.00 sshd
11062	paulbli	20	0	10936	1996	1468	S	0.0	0.0	0:00.00 bash
11086	root	20	0	52508	1832	1400	S	0.0	0.0	0:00.00 sudo
11087	root	20	0	10972	2124	1512	S	0.0	0.0	0:00.00 bash
11726	root	20	0	134208	5052	3820	S	0.0	0.1	0:00.01 sshd
11729	paulbli	20	0	134344	1996	756	S	0.0	0.0	0:00.34 sshd
11730	paulbli	20	0	10936	1996	1468	S	0.0	0.0	0:00.00 bash
11754	root	20	0	52508	1828	1400	S	0.0	0.0	0:00.00 sudo
11755	root	20	0	10972	2116	1504	S	0.0	0.0	0:00.02 bash
14414	root	20	0	37808	20236	8	S	0.0	0.2	0:00.87 secu-tcs-agent
18748	root	20	0	477040	4596	3420	S	0.0	0.1	0:23.57 agent
18766	root	20	0	20080	784	496	S	0.0	0.0	0:00.72 agentPlugInD
18787	root	20	0	23804	2108	900	S	0.0	0.0	0:26.45 base
18805	root	20	0	22272	992	784	S	0.0	0.0	0:01.59 tcvmstat
18838	root	20	0	9976	896	356	S	0.0	0.0	0:01.74 sysddd
19001	nsd	20	0	480376	1268	956	S	0.0	0.0	0:00.02 nsd
19156	root	20	0	64968	3756	2324	S	0.0	0.0	0:00.01 nsld
20091	root	20	0	53288	2144	1428	R	0.0	0.0	0:00.26 top
21818	root	20	0	7420	380	292	S	0.0	0.0	0:00.00 sleep
21971	root	20	0	7420	380	292	S	0.0	0.0	0:00.00 sleep
22015	root	20	0	7420	384	292	S	0.0	0.0	0:00.00 sleep
22879	root	20	0	120236	4576	1736	S	0.0	0.1	0:00.06 nginx
27973	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27974	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27975	nginx	20	0	120240	3456	280	S	0.0	0.0	0:00.00 nginx
27976	nginx	20	0	120240	3456	280	S	0.0	0.0	0:00.00 nginx
27977	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27978	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27979	nginx	20	0	120240	3460	284	S	0.0	0.0	0:00.00 nginx
27980	nginx	20	0	120240	3456	280	S	0.0	0.0	0:00.00 nginx
27981	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27982	nginx	20	0	120240	3456	280	S	0.0	0.0	0:00.00 nginx
27983	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27984	nginx	20	0	120252	3508	332	S	0.0	0.0	0:00.00 nginx
27985	nginx	20	0	120240	3456	280	S	0.0	0.0	0:00.00 nginx
27986	nginx	20	0	120252	3500	324	S	0.0	0.0	0:00.00 nginx
27987	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27988	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27989	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27990	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27991	nginx	20	0	120252	3508	332	S	0.0	0.0	0:00.01 nginx
27992	nginx	20	0	120252	3508	332	S	0.0	0.0	0:00.00 nginx
27993	nginx	20	0	120252	3512	336	S	0.0	0.0	0:00.00 nginx
27994	nginx	20	0	120252	3500	324	S	0.0	0.0	0:00.01 nginx
27995	nginx	20	0	120252	3508	332	S	0.0	0.0	0:00.02 nginx
27996	nginx	20	0	120252	3496	320	S	0.0	0.0	0:00.05 nginx

启动之后根据资源占用情况发现每台机器剩余内存不到7个g，所以对presto参数做了调整以保证他能拥有最大的使用率，由于coordinator上也设置了启动worker，那么我们现在可用的就是这两个节点。

下面是分别用Python多线程和Python多进程的测试结果

先贴代码：

```
def run_thread(arg):
    print('thread ' + str(arg) + " running...")
    conn = prestoedb.dbapi.connect(
        host='xxx',
        port=8080,
        user='xxx',
        catalog='xxx'
    )
    cur = conn.cursor()
    cur.execute("select * from xxx")
    begin = datetime.datetime.now()
    rows = cur.fetchall()
    end1 = datetime.datetime.now()
    time_interval = (end1 - begin).seconds
    print("producer time_interval:" + str(time_interval))
    print(len(rows))

if __name__ == '__main__':
    for i in range(10):
        run_thread('single'+str(i))
        # t = threading.Thread(target=run_thread, args=('threading'+str(i),))
        # t.start()
        # p = Process(target=run_thread, args=('process'+str(i),))
        # p.start()
```

由于资源有限，两台机子都是4核8G；分别对原生sql，优化后sql，使用单线程、多线程、多进程做了个比较：

未优化的sql花费时间

<u>Aa</u> Name	≡ 5次循环	≡ 10次循环	≡ 20次循环
<u>单线程</u>	25	50	100
<u>多线程</u>	13~14	25~28	56~60
<u>多进程</u>	8~11	17~19	25~30

优化后的sql花费时间

<u>Aa</u> Name	≡ 5次循环	≡ 10次循环	≡ 20次循环
----------------	--------	---------	---------

Aa Name	≡ 5次循环	≡ 10次循环	≡ 20次循环
<u>单线程</u>	10	20	40
<u>多线程</u>	8~9	16~18	28~30
<u>多进程</u>	6~7	11~13	16~19

总之从测试结果来看在资源耗尽的情况下多进程的效果仍旧是表现的最好的，但是在进程数到达一定数量后多线程的性能是会超过多进程的，因为这里只有IO请求没有任何计算，那么多线程的开销是要远远小于多进程的，我这里没法测试，因为我们Presto的服务器资源是有限的，过多的并发并不能反映最真实的结果。