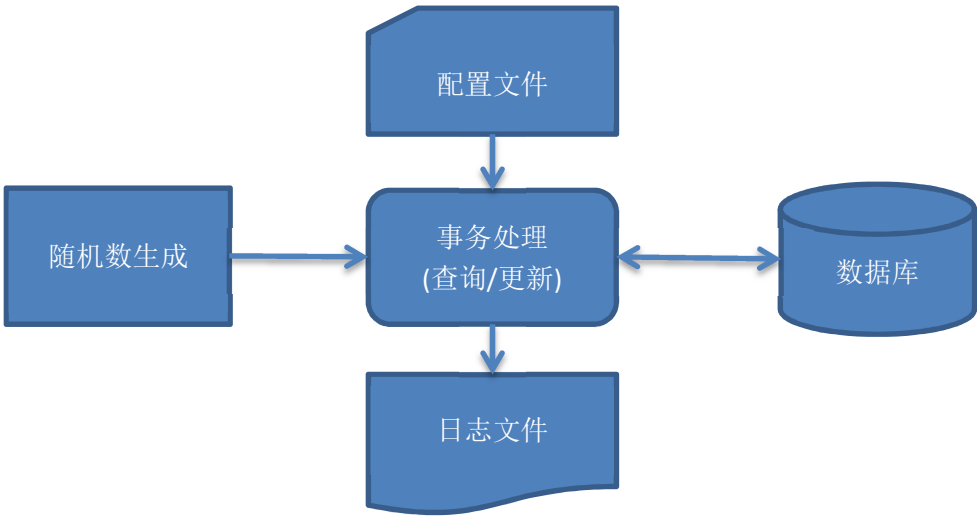
# 数据库压力测试工具

（微博：平民架构）

过去五年里，我在管理全球最大的在线支付系统（阿里巴巴集团旗下的支付宝）的数据库。2012 年 11 月 11 号全站支持了超过 1 亿笔的在线支付交易，核心数据库当天处理了 40 亿个数据库事务、285 亿次 SQL 执行、1930 亿次内存数据块访问、生成了 15GB 的数据库日志。面对这种压力，我需要精确地了解每个数据库（不管是 Oracle 还是 MySQL）所能支撑的业务能力，以便有足够的信心以支持 0 点时的高峰压力，高峰压力可能会达到平时的 6 倍以上。因此我花费了大量的时间在数据库的容量测试上，比如测试不同的硬件表现，例如：SSD、Fusion IO 等等，并且编写了自己的数据库压力测试工具，因为市面上找不到足够简单实用的同类工具。

测试中最重要的是对业务模型的抽象和测试，最好能知道不同压力下的 SQL 响应时间曲线，以避免系统雪崩效应。测试工具应当能使用你自己创建的业务表，很协助你很容易生成随机的压测数据，并且能定制关键的测试逻辑，并能生成足够详细的时延信息。然后可以结合应用对数据库层的时延要求，就可以知道高峰时段我们到底需要准备多少台数据库的主机，以避免巨大的资源浪费。

测试工具主要集中在事务能力方面，下图是压测工具的设计图（两年前画的图，让同事实现过一个版本，现在自己重新实现了一遍）。




后面会解释每一块的内容，然后你就会学会如何使用，可以用来测试数据库（不同软硬件条件下）的性能，以及进行容量评估，可以充分享受数据库压力测试的过程，不需要任何

脚本编程能力。

# 随机数生成

测试工具可以定义一些变量，并且设置随机值的取值范围，然后在 SQL 语句中用冒号加变量名来加以引用，Oracle 和 MySQL 都是同样的使用方式。例如：

*INSERT INTO <table name> (…) VALUES (:varname, :varname,…);*

*SELECT * FROM <table name> WHERE … > :varname;*

总共有 9 种不同的变量类型可以选择，变量定义的语法如下所示：

*变量名 VARTYPE 最小值 最大值*

*变量名 VARTYPE 值列表*

变量类型可以是“SEQ”、“INT”、”INTLIST”、“CHAR”、“STRLIST”、“FLOAT”、“DOUBLE”、“DATE”、“TIMESTAMP”中的任何一种，下面是各种变量类型的含义：

- SEQ

  自动递增的 32 位整数，从最小值开始一直到最大值，如果测试用例执行的次数超过最大值，则会自动从头开始循环使用，有点象 Oracle 数据库里的 Sequence 对象。例如：

  V_seq1 seq 1 100000000

  V_seq2 seq 1 100000000

- INT

  自动递增的 32 位整数，在最小值和最大值之间自动选取一个随机值。例如：

  V_int1 INT 1 100000000

  V_int2 INT 1 100000000

- INTLIST

从给定的值中自动选取一个整数值，多个值之间用逗号分隔，两个值中间不能带空格。例如：

V_int1 INTLIST 1,2,3,4,5,6,7

V_int2 INTLIST 1,1,2,2,3,3,4,4,5,5,6,6,7,7

- CHAR

字符串类型，自动生成值（A-Z 之间的字母填充），需要定义最小长度和最大长度，最大长度是 255 个字节。例如：

V_str1 CHAR 10 20

V_str2 CHAR 4 4

- STRLIST

从给定的字符串中自动选取一个字符串，多个字符串之间用逗号分隔。要求字符串的值不能包含空格，两个字符串之间不能带空格。例如：

V_str1 STRLIST Sun,Mon,Tue,Wed,Thr,Fri,Sat

V_str1 STRLIST Male,Female

- FLOAT

低精度浮点数，在最小值和最大值之间自动选取一个随机值。例如：

V_flt1 FLOAT 1 100000000

V_flt2 FLOAT 1 100000000

- DOUBLE

高精度浮点数，在最小值和最大值之间自动选取一个随机值。例如：

V_dbl1 DOUBLE 1 100000000

V_dbl2 DOUBLE 1 100000000

- DATE

自动生成的具有日期格式（YYYY-MM-DD）的字符串，需要指定日期范围，开始日期为当前日期减最小值，结束日期为当前日期加最大值。例如指定值“-10”表示

开始日期为 10 天前，指定”10”表示 10 天后。例如：

V_date1 DATE -10 10

V_date2 DATE -30 30

- TIMESTAMP

  自动生成的具有日期格式（YYYY-MM-DD HH24:MI:SS）的字符串，需要指定日期范围，开始日期为当前日期减最小值，结束日期为当前日期加最大值。例如指定值“-10”表示开始日期为 10 天前，指定”10”表示 10 天后。例如：

  V_datetime1 TIMESTAMP -10 10

  V_datetime2 TIMESTAMP -10 10

使用不同的变量类型，可以很容易模拟真实的数据分布情况来生成测试数据，即使不从生产库拖数据也可以精确地测试数据库性能。

# 配置文件

配置文件包含除并发线程数之外的所有测试信息，可以用任何文本编辑器来创建它，主要包含以下几部份：

**OPTION**

- 控制选项

**DECLARE**

- 变量定义

**BEGIN**

- SQL 语句

**END**

语法本身很象 Oracle 的 PL/SQL 代码块，因为对 Oracle 比较熟悉故采用这个格式。

# 控制选项

可以配置总共 10 个测试选项，下面是各个选项的具体介绍：

- user

  数据库连接信息，格式（用户名/口令@主机 IP:端口:数据库名或实例名）, Oracle 和 MySQL 都用这个统一的格式，DBA 应当对这个信息非常了解。

- loop

  测试用例的执行次数，默认值为 1 亿次，在生成测试数据时很有用，其他时侯建议用时间来控制。

- name

  压测用例名，仅仅显示在压测工具生成的日志文件里，方便记忆之用。

- wait

  两次执行（所有定义的语句被执行算一次）之间的暂停时间，单位为 0.1 毫秒，不是指两个 SQL 语句之间的暂停时间。默认值为 0 表示没有任何停顿，用大并发测试时需要指定一个暂停时间，若未定义则在超过 1000 个并发线程压测时会计算一个默认值。

- log

  输出的日志文件的名称，如果未定义则输出到当前屏幕，文件名中可以用“%p”来表示线程号，以便每个并发线程创建一个独立的日志文件。如果不想输出任何信息，则可以定义为”/dev/null”（非 Windows 平台）或”nul”（Windows 平台）。

- show

  时延信息报告频率，默认值是 300，即每隔 5 分钟输出一段测试报告。

- psmode

使用绑定变量模式，此选项只针对 MySQL 版本，默认值为关闭（NO），在 MySQL 上执行时所有的变量会被进行宏替换处理；如果设置为（YES），则使用绑定变量的接口来进行参数传递。

- tran

  事务模式开关，默认值为（OFF），即运行在自动提交模式，每一个 SQL 会被当作一个事务，执行成功则提交；如果设置为打开（ON），则第一条 SQL 执行之前会有一个开启事务的调用，所有 SQL 执行完成后，会有一个事务提交命令。

- commit

  事务大小参数，默认值为 1，每执行一次测试发一次提交命令，在测试中我们通常将一个事务的 SQL 写在一起，这个值是很合适的；当我们在造测试数据时，不妨调大一些到 100 或 1000，以加快测试数据生成过程，此参数只在事务模式下有效。

- time

  指定测试时间，默认值为 3600，即 1 小时。可以用”m”或”h”来简化时间指定。比如”5m”表示 5 分钟，“2h”表示两小时。

下面是一个完整的控制选项例子

***option***

*user <user>/<pass>@<ip>:<port>:<dbname>*

*loop 500000*

*log  select_userview_16k_%p.log*

*name Random_Userview_Select*

***declare***

# 变量定义

变理定义决定了随机值的生成方式，SQL 里用到的所有的变量都必须在这里事先定义好，在每执行一次测试前，会自动生成一次值，不是在每个 SQL 执行前随机生成一次，这样多个 SQL 之间的变量值可以共享，以进行真实的业务逻辑测试。可以参考随机数生成章节了解如何定义不同类型的变量。

一个完整的变量定义如下所示：

```
declare
  uid1 int 10000000 11000000
  uid2 int 10000000 11000000
  uid3 int 10000000 11000000
  uid4 int 10000000 11000000
  uid5 int 10000000 11000000
```

如果 SQL 里引用了很多变量，必须每个变量都进行定义，未定义的变量将以 NULL 值进行处理。

# SQL 语句

这里可以包含所有的 SQL 类型，包括 SELECT、INSERT、UPDATE 和 DELETE 语句，也可以用“{“和“}”将一段存贮过程括起来（见 Oracle 数据库），不同的语句之间用分号分开。

```
begin
  select col1 from T_KC_CENTER where col1 = :id;
  select col1, col2 from T_KC_CENTER where col1 = :id;
  select * from T_KC_CENTER where col1 = :id;
  { begin update t_kc_center set col2=col2 – 1 where col1 = :id;
        update t_kc_center set col2=col2 + 1 where col1 = :id + 1; end; }
end;
```

如果你希望 SQL 不是每次都执行，而是以特定的概率执行，可以用”RANDOM n”来控制执行比例，”n”指的是 1 和 100 之间的一个比例值。例如：

```
begin
   select col1 from T_KC_CENTER where col1 = :id;
   RANDOM 50 select col1, col2 from T_KC_CENTER where col1 = :id;
   RANDOM 20 select * from T_KC_CENTER where col1 = :id;
end
```

如果你希望用一个查询（通常是返回单行的 SQL 语句）来为变量赋值，只需要在 SQL 语句前加上”SETVAR”关键字，就可以将变量值放到与返回列同名的变量中了。例如：

```
begin
   SETVAR select dbms_random.value ID from dual;
   select col1, col2 from T_KC_CENTER where col1 = :id;
   select * from T_KC_CENTER where col1 = :id;
end
```

下面是一个完整的 MySQL 测试用例文件（测试工具和 MySQL 运行在同一台机器上）：

```
option
   name mysql_test
   loop 2000
   user /@::test
declare
   a int 20000 30000
   b int 20000 30000
begin
   select * from t_mytest where col1 = :a;
```

```
    random 50 select * from t_mytest where col1 = :b;
end
```

希望你会喜欢这种风格。

## 事务控制

使用“tran”控制选项，可以将所有的 SQL 语句放到一个事务中执行，或者运行在自动提交模式下。在真实的测试中我们还需要部份 SQL 能运行在自动提交模式下，部份 SQL 则可以自由组合成一个事务来执行。需要有一个机制来自定义事务的开始和结束。

可以使用“start”来显示开启一个事务，用“commit”或“rollback”来结束一个事务。例如：

```
option
    name mysql_test
    loop 2000
    user /@::test
declare
    a int 20000 30000
    b int 20000 30000
begin
    start;
    select * from t_mytest where col1 = :a;
    random 50 select * from t_mytest where col1 = :b;
    commit;
end
```

希望你会喜欢这个功能，它很容易模拟业务逻辑，并且能让你更加享受这个测试工具的便捷性。

# 运行测试用例

并行度只能通过命令行参数来指定，“query”选项来指定测试用例的配置文件，”degree”选项用来指定测试的并行度。用法如下：

./mydbtest_linux64.bin  query=配置文件 degree=8

例如:

./mydbtest_linux64.bin  query=test.cfg degree=8

最大的并行度可以是 16384，只需要一台测试机就可以制造足够的数据库压力。

# 日志文件

日志文件包含了每个 SQL 详细的时延信息，及执行次数（不同时延的分布）信息，同时也包括了更新的记录数等信息。下面是一个完整的日志文件例子：

```
ORADBTEST: Oracle Database Test Utility , Release 1.0.1
(@) Copyright Lou Fangxin (AnySQL.net) 2012 - 2013, all rights reserved.
======================== Oracle_Test ========================
SQL01   exe=5000 row=4999 ela=7570 ms avg=1514 us
SQL01       2 ms    exec=       4756, ela=          7018 ms, avg=       1475 us, pct= 95, 95
SQL01       3 ms    exec=        242, ela=           545 ms, avg=       2255 us, pct=  4, 99
SQL01       4 ms    exec=          2, ela=             6 ms, avg=       3253 us, pct=  0,100
Total   tran=5000=657/s, qtps=5000=657/s, ela=7573 ms, avg=1514 us
```

从上面的输出可以看到，平均SQL的时延是1.5 ms(1514 us)，单线程压到657个TPS。如果要估算总的 TPS 能力，只需要乘以并行度，在这时是 8，所以总的 TPS 能力在 5200 的样子。

你可以得到每个 SQL 的详细信息，“SQL01”表示第一个 SQL 语句（“SQL02”则表示第二个 SQL 语句，等等），大约 95%的请求是在 2ms 以内完成的，大约 4%的请求是在 2-3ms 内完成的，有两次请求的时延则超过了 3ms。

测试工具本身并不收集数据库的性能信息，请使用其他的工具来监控数据库的性能，并收集性能数据以进行分析。

## 获取软件

MySQL 数据库

http://www.mydul.net/software/mydbtest_linux32.zip

http://www.mydul.net/software/mydbtest_linux64.zip

Oracle 数据库

http://www.mydul.net/software/oradbtest.zip