HyBench

简介

Hybench 简介

Hybench 是一款由中国软件评测中心、清华大学联合牵头，北京奥星贝斯科技有限公司、武汉达梦数据库股份有限公司、华为技术有限公司、腾讯云计算有限公司、阿里云计算有限公司共同研发的 HTAP 数据库基准测试工具。

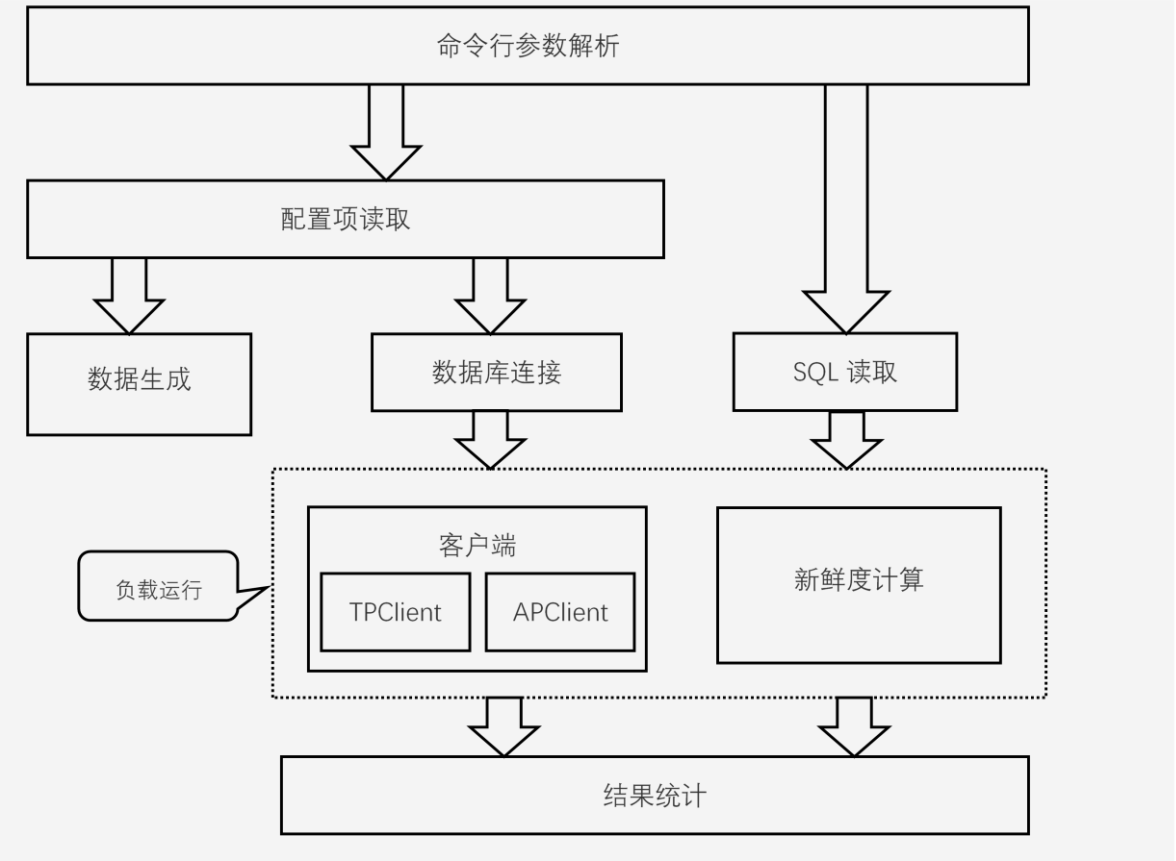
Hybench 针对 HTAP 数据库技术特点，结合实际典型应用场景进行设计，数据模型基于在线金融交易分析场景，提供 OLTP、OLAP、OLXP 三类典型 HTAP 负载，支持不同规模的数据集，可以计算出 TPS、QPS、XPS、新鲜度等不同维度的评价指标，最终给出统一评价指标 HTAP-Score。Hybench 旨在为数据库厂商和第三方测评机构提供 HTAP 数据库基准性能的评价方法及工具，以更好引导 HTAP 数据库技术研发方向，帮助用户进行 HTAP 数据库选型。

术语及缩略语

HTAP：Hybrid Transaction Analytical Processing，混合事务与分析处理
OLTP：Online Transaction Processing 联机事务处理
OLAP：Online Analytical Processing 联机分析处理
OLXP：Online miXed Processing 联机混合负载处理
混合负载：并发执行的交互式查询与分析型事务
TPS：Transaction Per Second 每秒事务处理数
QPS：Queries Per Second 每秒查询处理数
XPS：miXed operations Per Second 每秒混合负载处理数
新鲜度：主节点与副本节点的查询结果的最大时间延迟

系统概述

Hybench 实现了 Hybench 基准测试，集成了测试数据集生成，HTAP 负载运行、新鲜度度量、测试结果统计分析等多种功能，总体架构如下图所示：



通过该工具，我们不仅可以完成 Hybench 基准测试，得到评价指标 H-Score，还可以进行多种单项测试，对单项性能进行评判。下表是 Hybench 工具提供的测试类型以及对应的性能指标。

测试类型	性能指标
Hybench 基准测试	H-Score
OLTP 测试	TPS
OLAP 测试	QPS
OLXP 测试	XPS（包含 XP-TPS，XP-QPS）
新鲜度测试	Freshness

用户，数据库厂商，第三方评测机构均可使用 Hybench。用户可以利用 Hybench 进行基准测试，H-Score 可作为数据库选型的参考指标；数据库厂商在数据库研发过程中，可利用 Hybench 进行面向 HTAP 场景的性能评价和压测；第三方评测机构可利用 Hybench 提供客观的数据库性能测试服务。

总体设计

HyBench 程序可以分为 4 个大模块，分别是生成测试数据模块，参数与命令行解析模块，执行负载模块，结果收集与展示模块。其中执行负载模块又可以细化为分析型负载（AP），事务处理性负载（TP）以及新鲜度计算（Freshness）三个子模块。联机混合负载（XP）是融合了 IQ 和 AT 两种类型的负载，IQ 和 AT 分别在 AP 和 TP 模块中执行。另外分析型负载又分为的 power 测试和 throughtout 测试，throughtout 测试的结果会计算到 H-Score 中。

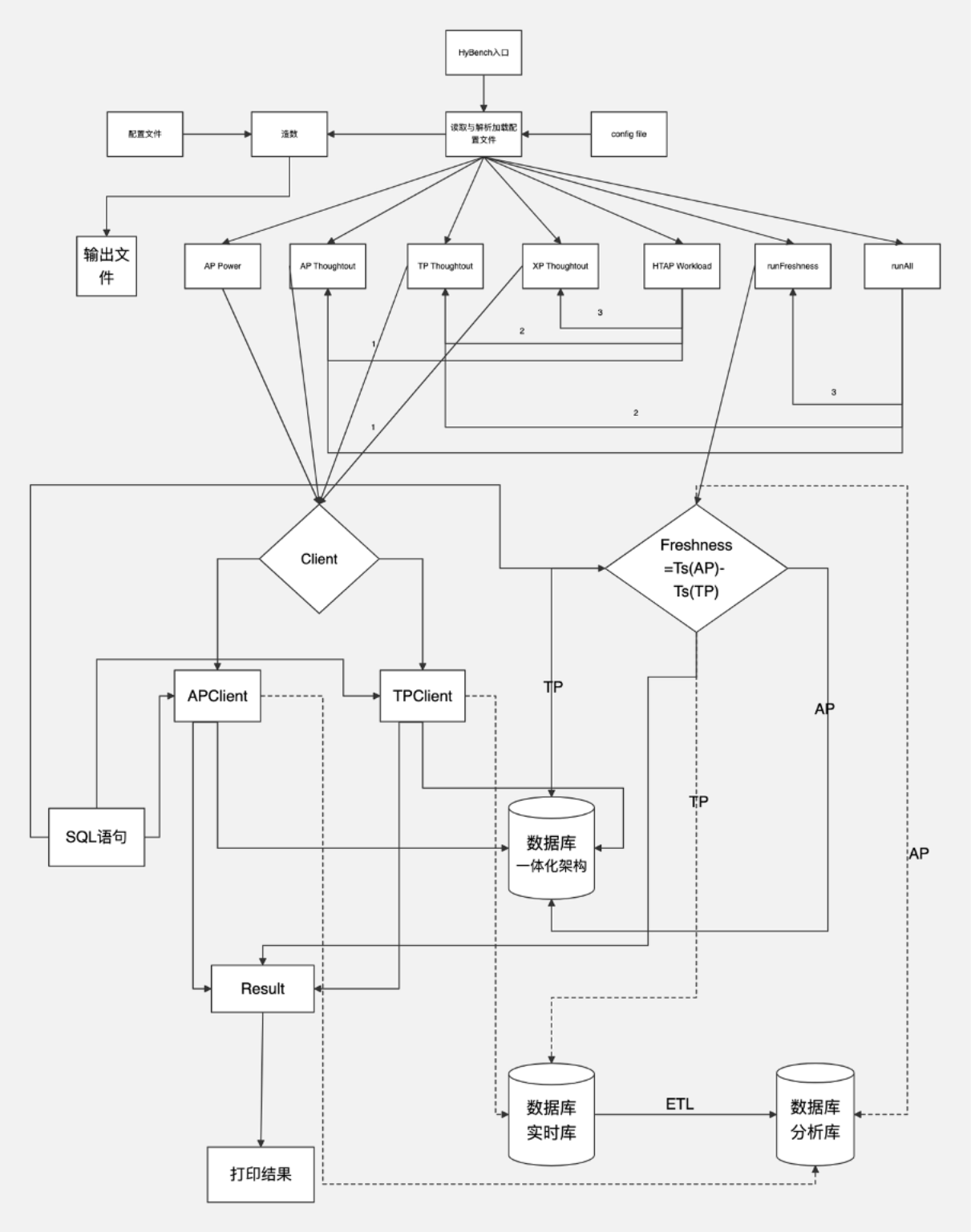
- 数据生成模块，可以通过配置文件配置数据量。各个特征数据的生成范围是记录在 resource/parameters.toml 中。
- 程序提供了多个执行选项，包括 gendata，runAPower，runAP ,runTP, runXP, runHTAP，runFresh 以及 runAll。通过解析模块解析命令行参数并加载配置文件执行相应的负载。
- 在执行负载模块中，APClient 和 TPClient 都继承 Client 类。Client 类中包括了负载执行时间的控制，负载并发控制，计算 TPS,QPS 等功能。APClient 和 TPClient 主要是执行各类 sql，统计事务执行时间等功能。
- 结果展示模块，主要用于执行期间的结果收集，以及最后 H-Scoure 的计算与各指标的输出。

Hybench 代码结构如下表所示

conf	用于存放 4 种配置文件 1. ap，tp 和 xp 的 sql file 2. 兼容各个数据库语法的 ddl 文件 3. 数据库连接信息，配置时间等配置文件 4. 日志配置文件	1. 提供了兼容 mysql、pg 和 oracle 三种语法规则的 sql 文件 Stmt_mysql.toml，stmt_oracle.toml，stmt_pg.toml 2. 提供了兼容 mysql、pg 和 oracle 三种语法规则的 ddl 文件 3. db.prop 4. log4j2.properties
lib	用于存放依赖的各类 jar	checker-qual-3.5.0.jar 、gson-2.8.1.jar log4j-core-2.19.0.jar commons-cli-1.4.jar guava-18.0.jar commons-

		math3-3.6.1.jar log4j-api-2.19.0.jar toml4j-0.7.2.jar
src	源代码文件，包括 java 文件以及 resource 文件	1. dbconn：数据库连接模块 2. load：数据生成模块 3. pojo：数据库表对应的对象 4. stats：结果统计与计算模块 5. util：随机数据生成模块 6. workload：包括 AP,TP 和新鲜度计算等执行模块 7. 其他：包括 main 入口，命令行解析，配置文件读取模块
hybench	执行程序的 Shell 脚本	里面配置了环境变量，封装了 java 执行命令

Hybench 接口调度如下图所示。



（虚线的指向是表示另外一种数据库系统，即通过 ETL 来同步实时事务系统产生的数据到分析系统。）

- **数据生成模块**

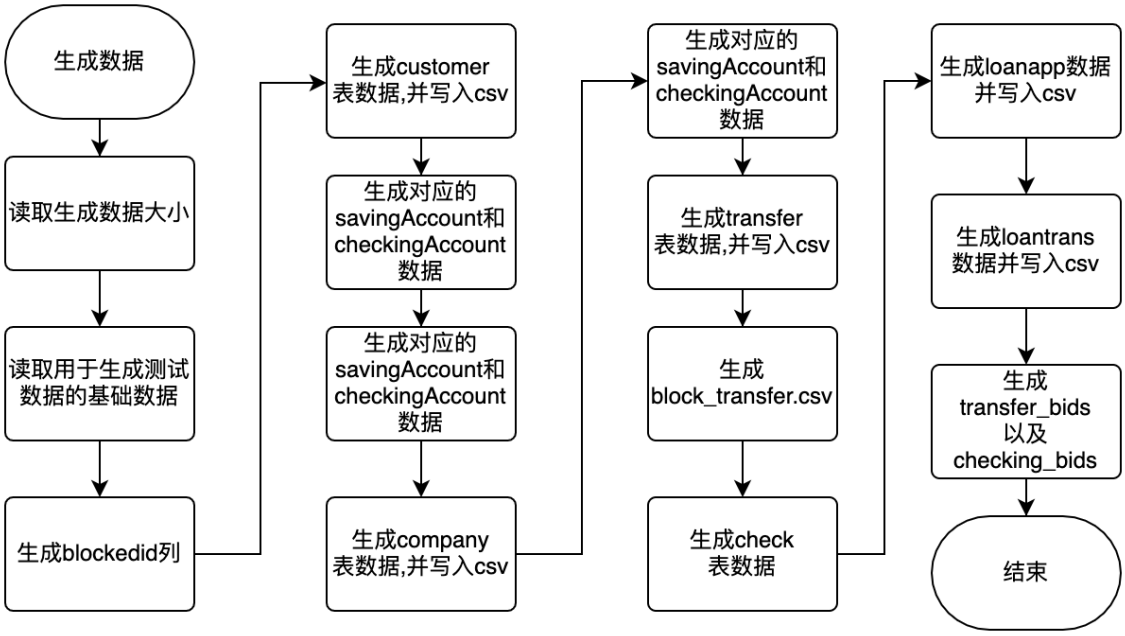
实现机制：

数据生成模块主要是用于生成测试时所需要的表数据,并以 csv 格式保存到磁盘上,以便数

据库进行导入操作.其在生成数据时,主要需要 db.prop 文件以及 parameters.toml 文件,在 db.prop 文件中主要需要读取的是 “sf” 参数,这个参数决定了生成的数据大小,之后数据生成模块会根据 sf 这个参数到 parameters.toml 读取对应的一些基础数据,这些基础数据包括需每个表需要生成的数据条数、每个列的起始以及结束值、生成数据文件的文件名等,之后数据生成模块会基于这些基础数据来生成测试所需要的数据文件.最后生成的数据文件如下所示:

```
~/ft_local/Data_1x
❯ ls                                                                                                    root
blocked_checking.csv  checkingAccount.csv  company.csv   loanApps.csv   Related_checking_bids  savingAccount.csv
blocked_transfer.csv  checking.csv         customer.csv  loanTrans.csv  Related_transfer_bids  transfer.csv
```

具体的,其生成数据的流程如下所示;



执行方法：

请注意，生成数据需要磁盘有足够的空间，否则会因为空间不足而导致报错退出。

1、首先修改 db.prop 文件，将 sf 修改为需要测试的数据集大小，如 1x 代表总 1GB 数据，10x 代表 10GB 数据。

2、然后执行下列语句，等待数据生成。

```
    ./hybench -t gendata -c conf/db.prop
```

3、完成数据生成执行后，可在 xxx 目录中查看到对应的数据文件。

contention_num 参数: 控制生成的 Related_transfer_bids 以及 Related_checking_bids 的大小,默认值为 100;

生成的数据文件路径：生成数据的文件在当前 HyBench 文件同级目录下.根据 sf 大小,文件夹名分别为 Data_1x,Data_10x,Data_100X,Data_1000x.

生成的 risk 回滚文件路径:文件路径同上

工具入口，配置项读取与命令行参数模块

工具的入口函数在文件 src/main/java/com/hybench/HyBench.java HyBench 类的 main 函数。

生成数据命令实例

```
    ./hybench -t gendata -c conf/db.prop
```

执行 conf/load_mysql.sql 文件中的 sql

	-t runhtap	runAP() runTP() runXP(0)	apclient/tpclient/xtpclient/xapclient/apRunMins/tpRunMins/xpRunMins	● apclient 执行 ap 负载的并发度，>1 时为 ● xtpclient：XP 负载中执行 tp 负载的并发 ● xapclient：XP 负载中执行 ap 负载的并发 ● tpRunMins：tp 负载的执行时间。 ● apRunMins: 在这时间段内会实时监控 qp 行结束时间为准。 ● xpRunMins: 在这时间段内会实时监控 qp 行结束时间为准。 ● apround: ap 负载执行的轮数，仅在 AP
	-t runxp	runXP(0)	xtpclient/xapclient/xpRunMins	
	-t runFresh	runFreshness(4)	xtpclient/xapclient/xpRunMins	
	-t runAll	runAP() runTP() runFreshness(4)	apclient/tpclient/xtpclient/xapclient/apRunMins/tpRunMins/xpRunMins	

- **数据库连接模块**

实现机制：

Hybench 通过引用对应数据库的 jdbc,通过 jdbc 链接从而访问到对应的数据库。

库存放位置：

要添加指定数据库对应的 jdbc 版本一共有两种方法,第一种是通过修改其 pom.xml 文件,通过 maven 自动下载相应的 jdbc 的 jar 包,如下所示:

```
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.32</version>
        </dependency>
```

第二种方法则是直接将 jdbc 的 jar 包拷贝到源代码的目录下,直接在 idea 中右键 jar 包,add as library 来添加

配置修改方法：

连接数据库的方法十分简单,只需修改 db.prop 文件中 jdbc 相关的配置即可,主要需要修改的是四个参数,首先是 classname,这里需要填写的是 jdbc 的具体类名,之后是数据库的 username 以及 password,最后需要填写的是数据库的 url,这里可以根据下述模板来进行修改

```
url=jdbc:mysql://<IP>:<PORT>/<db>?useUnicode=true&characterEncoding=utf-8
```

这里需要注意的是,由于 tp 和 ap 本系统是分别进行链接的,因此需要分别配置 tp 的连接参数与 ap 的连接参数

- **task 抽象与 sql 读取模块**

task 抽象

我们根据负载执行的事务类型，将 Hybench 测试基准的客户端分为 2 类，TP 客户端和 AP 客户端。

TP 客户端（事务型负载）	联机事务
	分析型事务
AP 客户端（分析型负载）	联机分析
	交互式查询

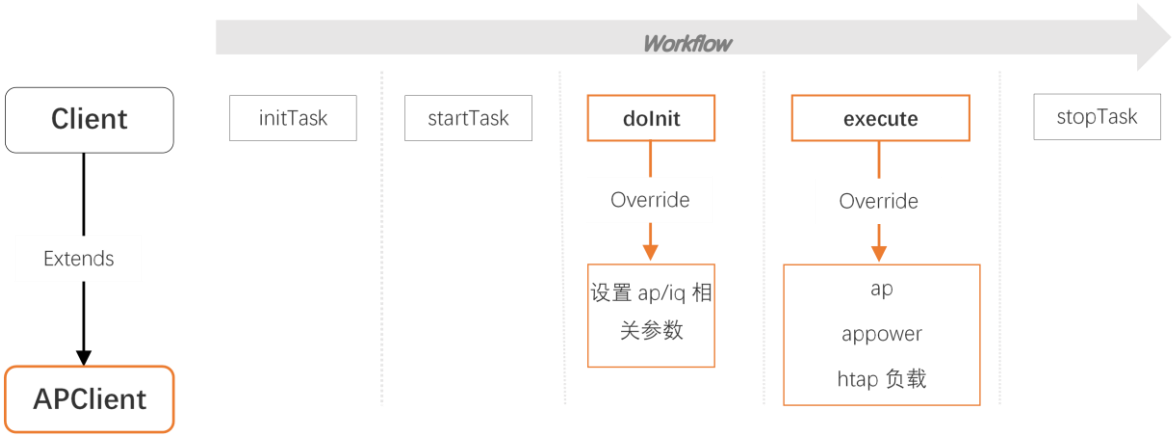
hybench 实现中，将这 2 种客户端都抽象为 Client 类。其实现 workflow 如下图。

sql 读取模块

为了方便数据库厂商进行 SQL 适配，hybench 实现了通过 toml 文件配置事务 SQL，其解析流程如下：

- **AP/IQ 执行模块**

AP/IQ 执行模块，对应 APClient，继承了 Client，重写了 Client 的 doInit 和 execute，实现 workflow 如下图。

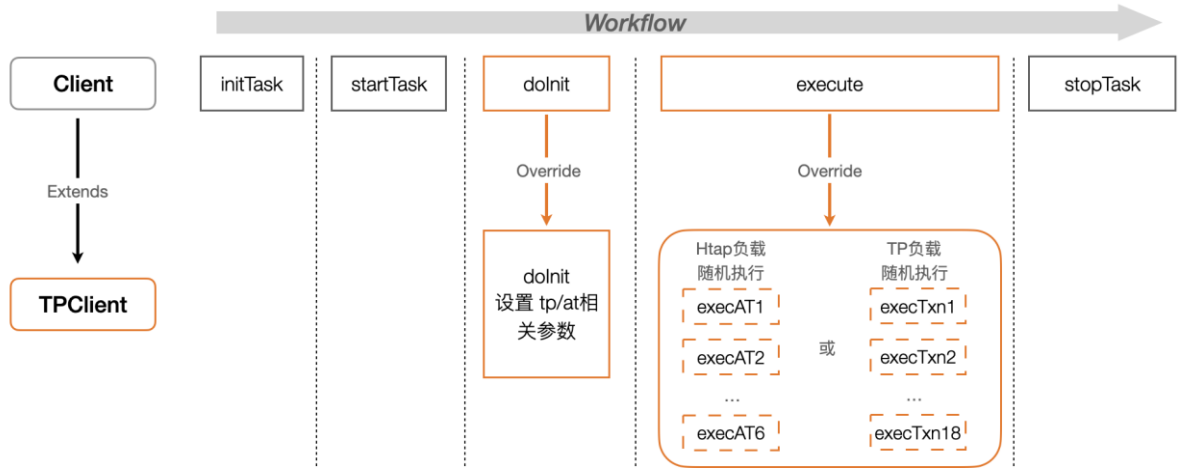


ap 提供的任务类型如下表所示：

ap	运行多轮 AP 查询。每轮都会将 13 个 AP 查询乱序执行。
appower	根据指定的轮数，依次运行 13 个 AP 查询。
htap	随机选择 IQ 查询运行

- **TP/AT 执行模块**

TP 和 HTAP 负载执行模块，对应 TPClient，继承了 Client，重写了 Client 的 doInit 和 execute，实现 workflow 如下图。



- **新鲜度计算模块**

新鲜度是基于 OLXP 负载测试的性能度量指标，描述了查询结果集数据中的最大时间戳与最新事务数据时间戳间的差距，理想情况下，查询操作总是能读到最新的数据，即查询结果时间与最新事务时间完全一致，此时性能达到最优。

根据上述描述，可将新鲜度的计算公式定义为：

$$\overline{f_T} = \sum_{Q \in W} (T_M - T_Q)/|W|$$

其中，TM 是最新事务数据的时间戳，TQ 是查询结果集 Q 的最大时间戳，W 是所查询的实例（OLTP 实例或 OLAP 实例）。

由于数据间同步往往有延迟，故新鲜度以整体数据查询最大延迟时间的平均值作为评价指标，以更好的评估 OLXP 负载的性能优劣。

- **结果统计模块**

HTAP 负载提供 OLTP、OLAP、OLXP 三类典型负载的测试，综合三个负载的评价指标，本节提出以 HTAP-Score 来衡量系统的混合事务分析处理性能，其计算公式为：

$$HTAP-score=\frac{\sqrt[3]{TPS*QPS*(ATS+IQS)}}{\overline{f_T}+1}@SF(n,m)$$

其中，每秒事务处理数（TPS）和每秒查询处理数（QPS）分别为针对 OLTP 和 OLAP 的评价指标，OLXP 负载选取分析型事务的每秒吞吐量（ATS）、实时性分析事务的每秒吞吐量（IQS）以及新鲜度 ft 共同做为评价指标；SF 为扩展因子，n 和 m 分别为 OLTP 与 OLAP 的并发线程数。

数据新鲜度以秒为量纲，新鲜度时间越大，整体性能越差

使用说明

使用说明描述 hybench 的如何运行，依赖有哪些。

准备测试工具：

下载 Hybench 负载测试程序 HyBench-1.0.jar，准备好测试所需的 jdbc 驱动程序，按照第三节使用说明在 HyBench 中引用驱动。

生成配置文件：

测试运行配置文件为测试程序目录下的 db.props。根据待测数据库的实际情况，修改配置项对应的参数内容，主要包括：

db：测试数据库种类

classname/classname_ap：TP/AP 类访问连接使用的 JDBC 类

username/username_ap：TP/AP 类访问连接的使用的用户名

password/password_ap：TP/AP 类访问连接的使用的密码

url/url_ap：TP/AP 类访问连接的 jdbc 连接字串

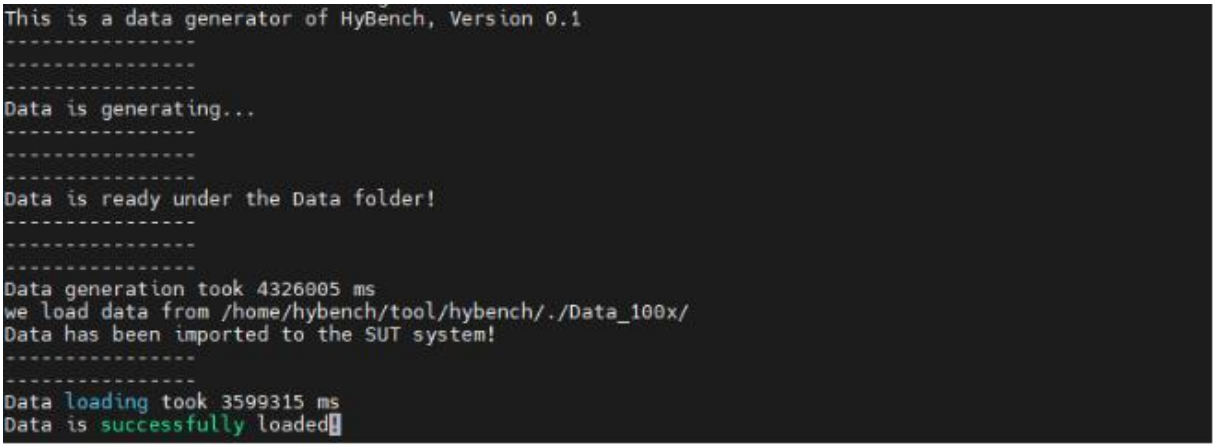
sf：数据规模，代表测试的数据集大小

参数详细介绍及其他可选参数配置项可参考“参数列表”一节。

创建并导入测试数据：

```
jar ./HyBench-1.0.jar -t load -c conf/db.prop
```

通过-t 指定 load 参数，进行创建数据并导入数据库的操作，-c 指定配置好的 props 文件。该指令会根据配置文件中指定的 sf 参数生成对应规模的数据，命令执行完成后，在同级目录下会生成以 sf 规格命名的文件夹；并根据配置文件中指定的 jdbc 连接字串连接数据库，将生成的数据装载到数据库中。



测试执行：

		字，若为 MySQL 协议的，则填写 MySQL,若为 postgresql 协议的则填写 postgresql，同时也可以填写其他类型的数据库。	义，主要用于打印结果。
classname	com.mysql.jdbc.Driver	指 TP 类的 SQL 访问采用那个 JDBC 的类	
username	xxx	指 TP 类访问连接的用户名	
password	xxx	指 TP 类访问连接的密码	
url	jdbc:oceanbase://<IP> :«PORT>/<db>?useUnicode=xxx	TP 类访问的 jdbc 连接串，根据各 jdbc 的定义自行填写。后续可根据实际情况填写连接参数	
url_ap	jdbc:oceanbase://<IP> :«PORT>/<db>?useUnicode=xxx	AP 类访问的 jdbc 连接串，根据各 jdbc 的定义自行填写。后续可根据实际情况填写连接参数	
classname_ap	com.mysql.jdbc.Driver	指 AP 类的 SQL 访问采用那个 JDBC 的类	
username_ap	xxx	AP 类访问连接的用户名	
password_ap	xxx	AP 类访问连接的用户名	
sf	1x	代表测试的数据集大小，仅支持 1x、10x	在生成数据和执行 SQL 时候均会用到。只能支持 2 个入参值，不支持其他的数据集大小。
at1_percent	35	代表在执行混合负载时，AT1 类	与其他几个 AT 类的

		的 SQL 的执行比例。如值为 35，则代表为 35%。	SQL 的执行比例合为 100。建议直接使用默认值。
at2_percent	25	代表在执行混合负载时，AT2 类的 SQL 的执行比例。如值为 25，则代表为 25%。	与其他几个 AT 类的 SQL 的执行比例合为 100。建议直接使用默认值。
at3_percent	15	代表在执行混合负载时，AT3 类的 SQL 的执行比例。如值为 15，则代表为 15%。	与其他几个 AT 类的 SQL 的执行比例合为 100。建议直接使用默认值。
at4_percent	15	代表在执行混合负载时，AT4 类的 SQL 的执行比例。如值为 35，则代表为 15%。	与其他几个 AT 类的 SQL 的执行比例合为 100。建议直接使用默认值。
at5_percent	7	代表在执行混合负载时，AT5 类的 SQL 的执行比例。如值为 35，则代表为 7%。	与其他几个 AT 类的 SQL 的执行比例合为 100。建议直接使用默认值。
at6_percent	3	代表在执行混合负载时，AT6 类的 SQL 的执行比例。如值为 35，则代表为 3%。	与其他几个 AT 类的 SQL 的执行比例合为 100。建议直接使用默认值。
apclient	1	在执行 AP 类的测试时，所开启的 AP 类连接数量。也可以理解为 AP 请求的并发数。	
tpclient	1	在执行 TP 类的测试时，所开启的 TP 类连接数量。也可以理解为 TP 请求的并发数。	
fresh_interval	20	新鲜度测试的间隔时间。基于 xpRunMins 的值来判断。若此值设置为 20.xpRunMins 设置为 1 分钟，则代表 1 分钟除以 20 为	

		3 秒。意味着每 3 秒执行以下新鲜度的查询。	
apRunMins	1	AP 类测试的运行时长。	
tpRunMins	1	TP 类测试的运行时长	
xpRunMins	1	XP 类测试的运行时长	
xtpclient	1	在 XP 测试场景中，TP 类型的请求所使用的连接数。	
xapclient	1	在 XP 测试场景中，AP 类型的请求所使用的连接数	
around	1	AP 类型请求的执行轮数，要求至少 1 轮完整跑完。	仅在 AP Power 测试负载中生效。

结果说明

本节会对 hybench 各个负载测试有意义的输出结果进行说明。

TP 测试负载 - runtp

1. TPS ：统计了整个执行时间段内的平均 TPS，反映了整个系统的绝对性能。
2. TP 事务的延迟数据统计信息，单位 ms，如下。依次表示 TP 事务 9 的最大延迟，最小延迟，平均延迟，95%分位延迟，99%分位延迟。这条数据可以用于分析各个事务在被测系统是否存在性能瓶颈，帮助性能优化。

```
TP Transaction  9 : max rt :      58.00 | min rt :       1.00 |
avg rt :       2.30 | 95% rt :       3.00 | 99% rt :       3.00
```

AP 测试负载 - runap

1. total elapsed time : AP 负载的执行时间。
2. AP Query 延迟数据统计，单位 ms，示例如下。依次表示 AP Query 4 的最大延迟，最小延迟，平均延迟，95%分位延迟，99%分位延迟。这条数据可以用于分析各个 query 在被测系统是否存在性能瓶颈，帮助性能优化。

```
AP Query  4 : max rt :       3.00 | min rt :       1.00 | avg
rt :       2.38 | 95% rt :       3.00 | 99% rt :       3.00
```

XP 测试负载 - runxp

1. XP-QPS 和 XP-TPS: 分别表示 XP 负载的 QPS 和 TPS。
2. 各个 Interative Query 和 Analytical Transaction 的延迟信息，用于分析各个 query 和事务在被测系统是否存在瓶颈。

新鲜度测试 - runFresh

新鲜度测试的评估结果是 Freshness 。理想的情况是查询总是能读到最新的数据，即新鲜度为 0。然而如果是主副本异步更新或者是 OLTP 实例与 OLAP 实例分离等情况，由于数据同步有延迟，查询读到的数据不一定是最新的。针对新鲜度计算，由于直接扫描查询涉及的数据会对系统的性能造成较大的影响。因此，采取查询结果集的时间戳对比方式，以计算数据新鲜度。

AP Power Test - runappower

同 runap 测试的结果。

HTAP 测试负载 - runhtap

包含 runap、runtp、runxp 的测试结果，参照上文表述。

所有负载测试 - runAll

包含 runap、runtp、runFresh 的测试结果，参照上文表述。