PostgreSQL 数据库选型测试报告

一、MySQL & PostgreSQL 数据库特性对比

项目	MySQL 	PostgreSQL 
适用场景	OLTP	OLTP/OLAP
开源协议	GNU General Public License (GPLv2)	PostgreSQL is released under the PostgreSQL License, a liberal Open Source license, similar to the BSD or MIT licenses.
流行程度	目前最流行的开源数据库，DB-Engines 网站排名第二 2019 年度数据库	DB-Engines 目前排名第四，上升趋势非常明显，全球最先进的开源数据库。 2017、2018、2020 年度数据库
线程模型	单进程多线程	多进程
主从复制	Binlog 逻辑复制 逻辑复制，延迟大(大数据量 dml、load data 等) 逻辑复制容易主从数据不一致	物理复制 物理复制，延迟小 物理复制，主备数据严格一致
优化器	基于 CBO 的优化器 统计信息简单 8.0 之前只支持 nest loop join 复杂查询性能不佳	功能更完备的 CBO 优化器 统计信息丰富 支持 nest loop,merge sort 和 hash join 复杂查询性能很好，支持序列、窗口函数、CTE 等
并行查询	不支持并行查询	支持并行查询
分区表	Mysql 分区表问题多，通常规范中不推荐使用	分区表功能成熟，大量用于生产系统
可扩展性	PostgreSQL 是高度可扩展的，您可以添加和拥有数据类型，运算符，索引类型和功能语言。	无法扩展
高可用方案	依赖第三方工具， 如 mha/orachestrator 等	依赖第三方工具， 如 patroni/repmgr/pg_pool 等
分布式方案	mycat	Citus/pgxc/pgxl Greenplum/Yellobrick/TBase/GaussDB/
总结	MySQL 是单进程多线程的开源数据库，流行程度高，适合高并发、中小规模数据量下的 OLTP，多表关联性能很低，通常单表数据量不超过 500 万行，多表关联通常不能超过 3 个表。数据量大之后需要分库分表但有关联问题。	PostgreSQL 是多进程的开源数据库，流行程度比 MySQL 低，完全开源，功能强大，优化器成熟。能够处理 OLTP 和 OLAP 的 HTAP 数据库，大数量和多表关联下仍然有很好的性能，分区表功能完善。 PostgreSQL 更接近 Oracle，都是多进程、物理复制、支持并行查询，是很多公司脱 O 迁出的首选数据库。
适用场景	业务形态简单、高并发、数据量不大的 OLTP	中大规模数据量的 HTAP 复杂查询、多表关联、分析函数等的 OLAP Oracle、DB2 等商业数据库迁移首选数据库

二、MySQL & PostgreSQL TPC-C/TPC-H 压测

2.1 环境配置

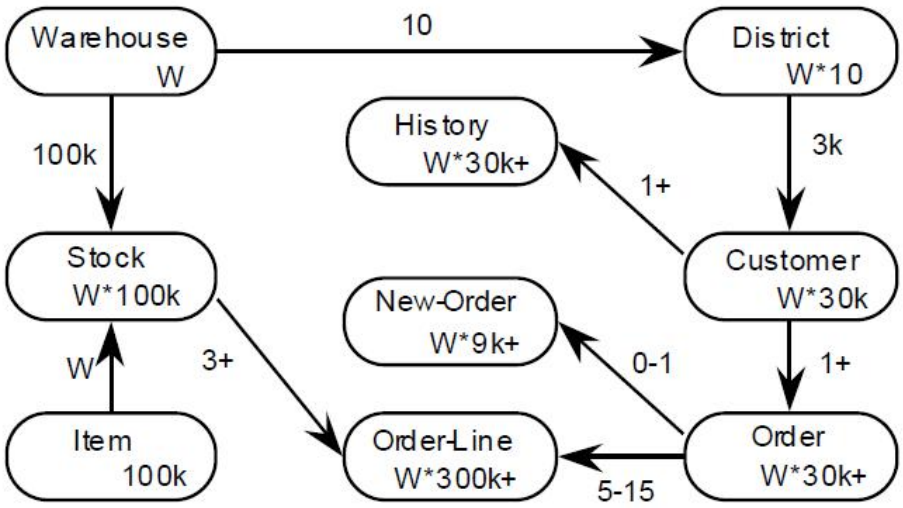
类别	说明
OS	Docker CentOS 7.6-1.3.0 镜像
CPU	32C
Memory	180G
存储	8TB
MySQL 版本	5.7.25
PostgreSQL 版本	12.5

2.2 专业术语

TPC: Transaction Processing Performance Council,是一个非盈利组织，成立于 1988 年，这个组织主要的功能是定义事务处理、数据库的基准，这个基准用于评估服务器的性能，并且把服务器评估的结果发布在 TPC 的官方网站上。简单的讲 TPC 就是系统评测皇家科学院。

TPC-C: TPC-C is an On-Line Transaction Processing Benchmark.
TPC-C is measured in transactions per minute (tpmC).
TPC-C 是业界常用的一套 benchmark，由 TPC 委员会制定发布，用于评测数据库的联机交易处理（OLTP）能力。主要涉及 10 张表，包含五类业务事务模型（NewOrder－新订单的生成、Payment－订单付款、OrderStatus－最近订单查询、Delivery－配送、StockLevel－库存缺货状态分析）。

TPC-C 事务模型
TPC-C 需要处理的交易事务主要为以下几种：
1、新订单（New-Order） ：客户输入一笔新的订货交易；
2、支付操作（Payment） ：更新客户帐户余额以反映其支付状况；
3、发货（Delivery） ：发货（模拟批处理交易）；
4、订单状态查询（Order-Status） ：查询客户最近交易的状态；
5、库存状态查询（Stock-Level） ：查询仓库库存状况，以便能够及时补货。



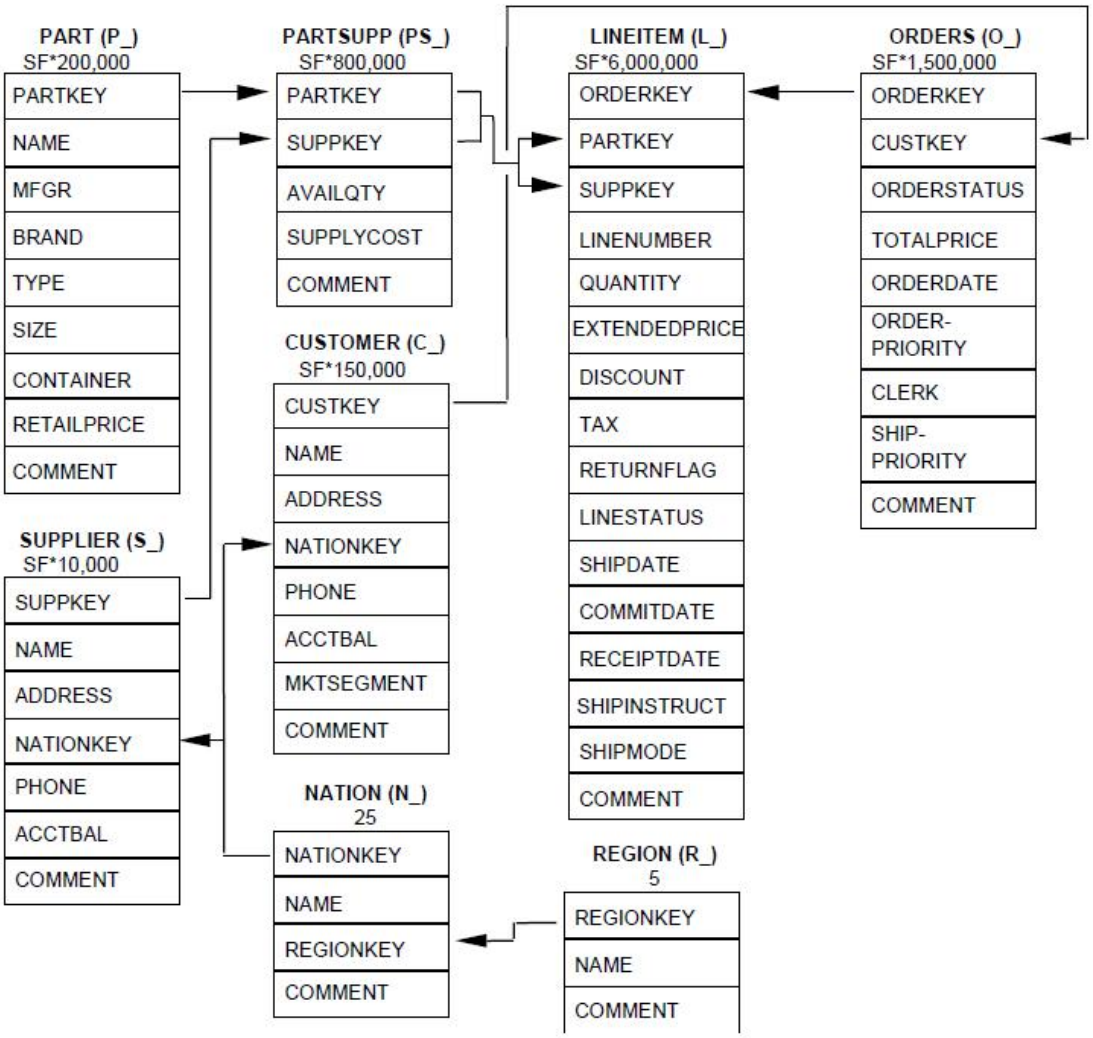
TPC-C 通过 tpmC 值（Transactions per Minute）来衡量系统最大有效吞吐量（MQTh，Max Qualified Throughput），其中 Transactions 以 NewOrder Transaction 为准，即最终衡量单位为每分钟处理的新订单数

TPC-C 性能衡量指标 tpmC
流量指标(Throughput,简称 tpmC)：按照 TPC 组织的定义，流量指标描述了系统在执行支付操作、订单状态查询、发货和库存状态查询这 4 种交易的同时，每分钟可以处理多少个新订单交易。所有交易的响应时间必须满足 TPC-C 测试规范的要求，且各种交易数量所占的比例也应该满足 TPC-C 测试规范的要求。在这种情况下，流量指标值越大说明系统的联机事务处理能力越高。

TPC-H:
TPC-H is a Decision Support Benchmark
TPC-H 是业界常用的一套 Benchmark，由 TPC 委员会制定发布，用于评测数据库的分析型查询能力。TPC-H 查询包含 8 张数据表、22 条复杂的 SQL 查询，大多数查询包含若干表 Join、子查询和 Group-by 聚合等。
简单来说，TPC-H 就是通过 22 个复杂 SQL 查询来评估数据库 OLAP 的性能。

TPC-H 表模型：

Figure 2: The TPC-H Schema



2.3 测试场景

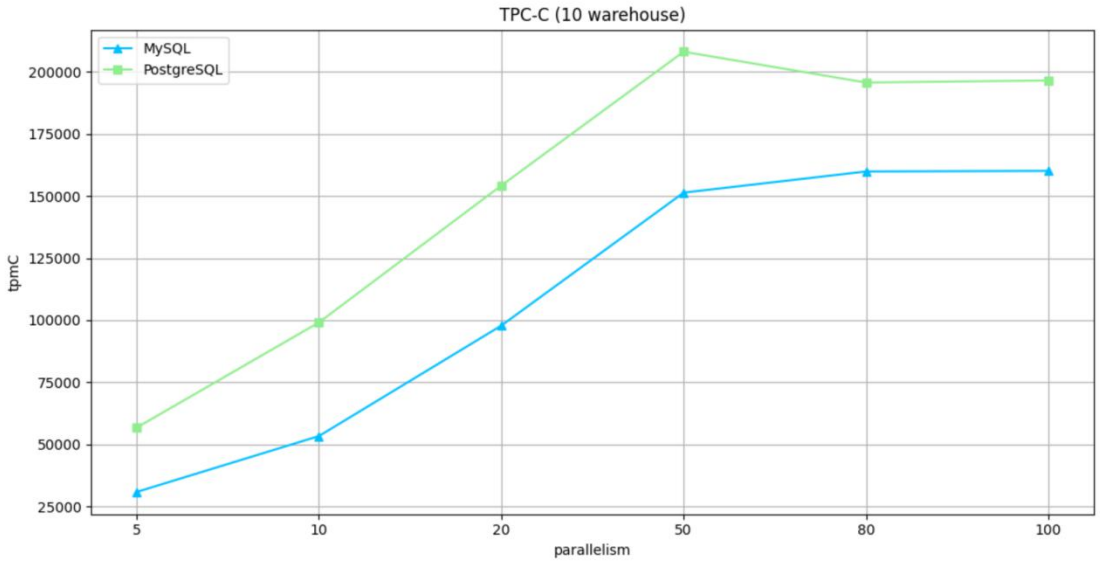
通过 BenchmarkSQL 的 TPC-C 模型压测 MySQL/PostgreSQL 在不同数据量(warehouse)不同并发度下的 OLTP 处理能力

TPC-H 压测 MySQL/PostgreSQL OLAP 处理能力

2.4 测试过程与测试结果：

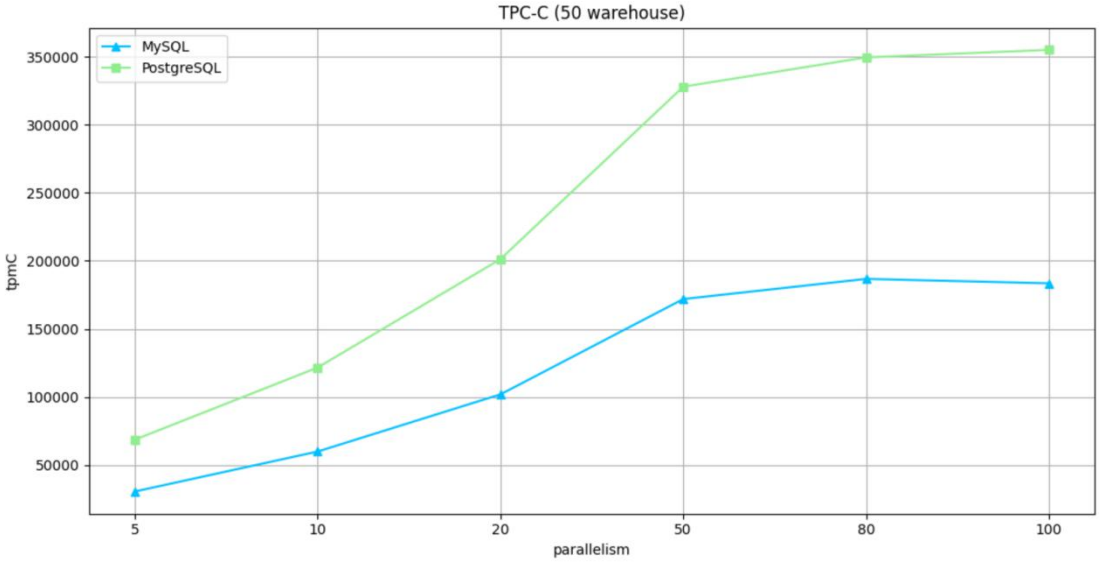
TPC-C 10 warehouse(数据量 customer 300000, 数据量 300 0000)

并发度	tpmC-MySQL	tpmC-MySQL(备注)	tpmC-PostgreSQL	tpmC-PostgreSQL(备注)
5	30872.68		56751.44	
10	53372.98		99111.11	CPU 16%
20	97824.88		154200.91	CPU 34.3%
50	151354.23	CPU 72%	208079.38	CPU 77.2%
80	159871.29	CPU 86%	195675.97	CPU 92%
100	160104.39	CPU 89.6%	196503.94	CPU 100%



TPC-C 50 warehouse(数据量 customer 150 0000, 数据量 1500 0000)

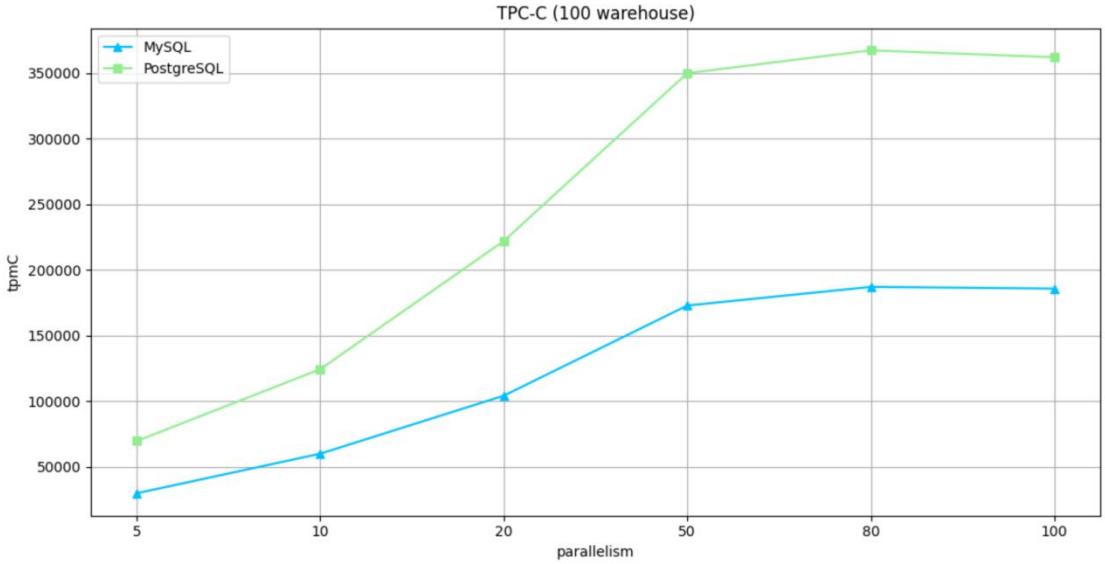
并发度	tpmC-MySQL	tpmC-MySQL(备注)	tpmC-PostgreSQL	tpmC-PostgreSQL(备注)
5	30630.98		68698.88	CPU 9.8%
10	59976.6	CPU 20%	121670.08	CPU 19.6%
20	101944.25	CPU 40%	201264.77	CPU 39%
50	171949.63	CPU 86%	327904.45	CPU 85.3%
80	186754.5	CPU 100%	349437.28	CPU 100%
100	183519.52	CPU 100%	354960.95	CPU 100%



TPC-C 100 warehouse(数据量 customer 300 0000, 数据量 3000 0000)

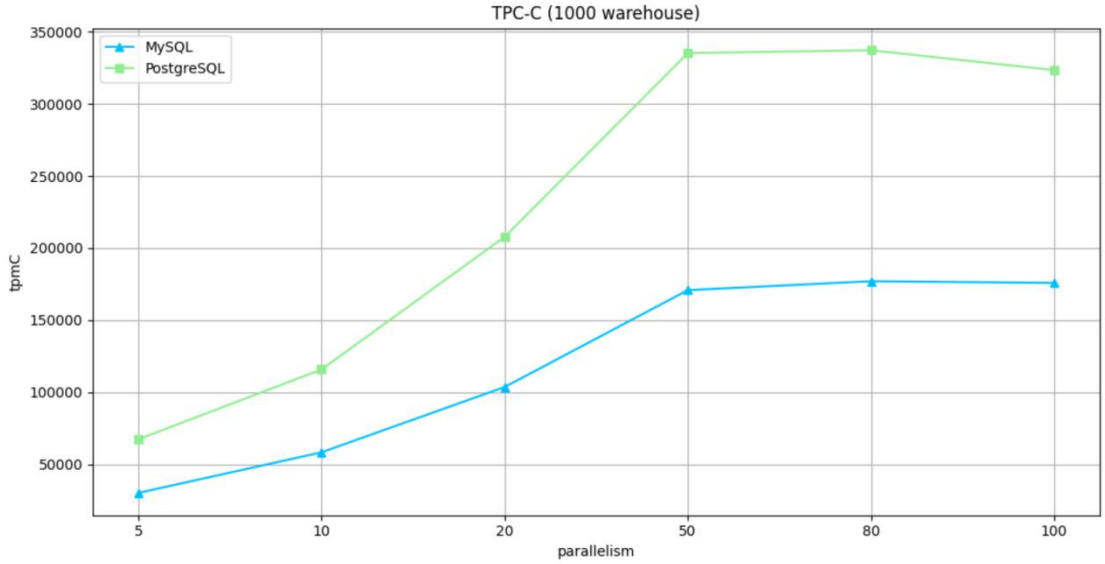
并发度	tpmC-MySQL	tpmC-MySQL(备注)	tpmC-PostgreSQL	tpmC-PostgreSQL(备注)
5	29656.35	CPU 10.5%	69456.03	CPU 9.7%

10	59816.2	CPU 21.2%	124215.02	CPU 24.2%
20	104020.01	CPU 40.8%	221906.19	CPU 40.2%
50	172759.54	CPU 87.5%	349877.03	CPU 87.8%
80	186974.33	CPU 100%	367292.42	CPU 100%
100	185676.76	CPU 100%	361991.76	CPU 100%



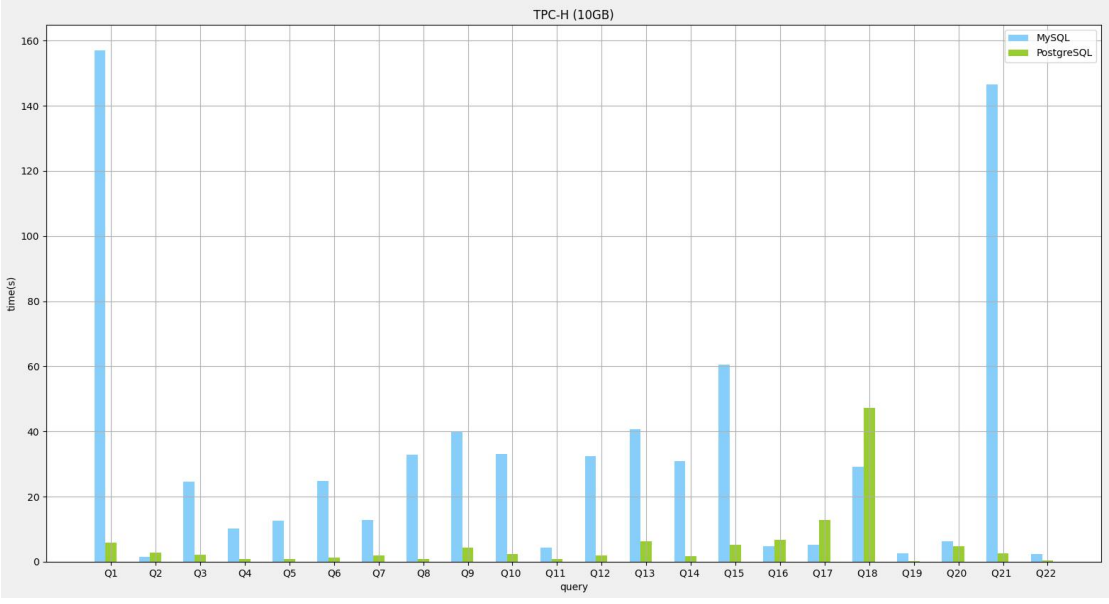
TPC-C 1000 warehouse(数据量 customer 300 0000, 数据量 30000 0000)

并发度	tpmC-MySQL	tpmC-MySQL(备注)	tpmC-PostgreSQL	tpmC-PostgreSQL(备注)
5	30121.84	CPU 10.5%	67347.31	CPU 9.3%
10	58283.22	CPU 20.8%	115771.29	CPU 19.6%
20	103594.32	CPU 41.6%	207786.85	CPU 37.9%
50	170775.55	CPU 86.8%	335236.92	CPU 86.6%
80	176972.6	CPU 93.6%	337200.53	CPU 93.7%
100	175798.1	CPU 92.8%	323425.16	CPU 92.3%



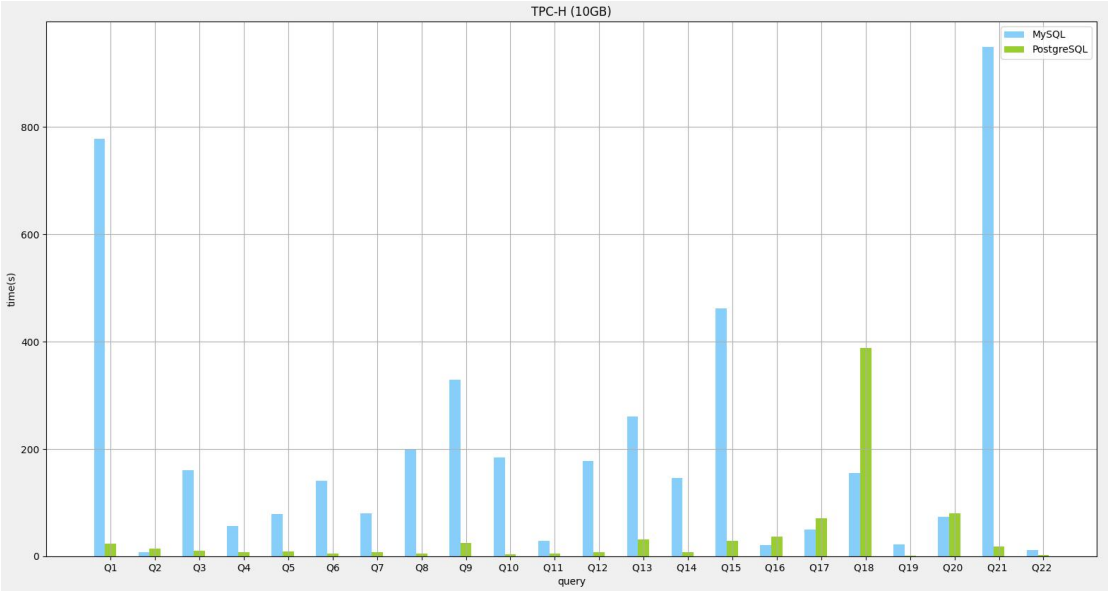
TPC-H 10G(Scalar Factor = 10),数据量(LINEITEM 表约 6000 万行，ORDERS 表 1500 万行，PARSUPP 表 800 万行)

Query	MySQL（单位：秒）	PostgreSQL（单位：秒）
Q01	157	5.90
Q02	1.53	2.89
Q03	24.60	2.15
Q04	10.12	0.80
Q05	12.51	0.92
Q06	24.90	1.33
Q07	12.81	1.91
Q08	32.76	0.86
Q09	40.08	4.39
Q10	33.16	2.40
Q11	4.34	0.77
Q12	32.32	1.84
Q13	40.67	6.28
Q14	30.95	1.67
Q15	60.45	5.24
Q16	4.69	6.63
Q17	5.28	12.75
Q18	29.14	47.30
Q19	2.55	0.17
Q20	6.30	4.67
Q21	146.52	2.59
Q22	2.33	0.44



TPC-H 50G(Scalar Factor = 50),数据量(LINEITEM 表约 3 亿行，ORDERS 表 7500 万行，PARSUPP 表 4000 万行)

Query	MySQL（单位：秒）	PostgreSQL（单位：秒）
Q01	778.42	23.42
Q02	7.35	14.23
Q03	160.87	10.00
Q04	56.07	7.80
Q05	79.27	8.88
Q06	140.67	4.87
Q07	80.00	7.88
Q08	198.67	5.68
Q09	329.18	25.29
Q10	184.32	13.25
Q11	28.16	4.86
Q12	178.12	7.87
Q13	259.97	31.60
Q14	146.12	7.60
Q15	462.34	28.50
Q16	20.87	37.07
Q17	50.52	71.14
Q18	155.30	388.95
Q19	21.66	0.95
Q20	73.77	80.04
Q21	949.47	18.15
Q22	11.64	3.12



附录：苏宁易购 MySQL/PG 压测数据对比

https://www.sohu.com/a/433657649_411876

2.5 测试结论：

TPC-C 的 OLTP 测试中，PostgreSQL 在不同数据量不同并发度下有更高 tpmC.
TPC-H 的 OLAP 测试中，PostgreSQL 在大部分复杂查询下比 MySQL 都有更低的响应时间，尤其是在大数据量以及多表关联的复制查询，PostgreSQL 表现出更好的性能。

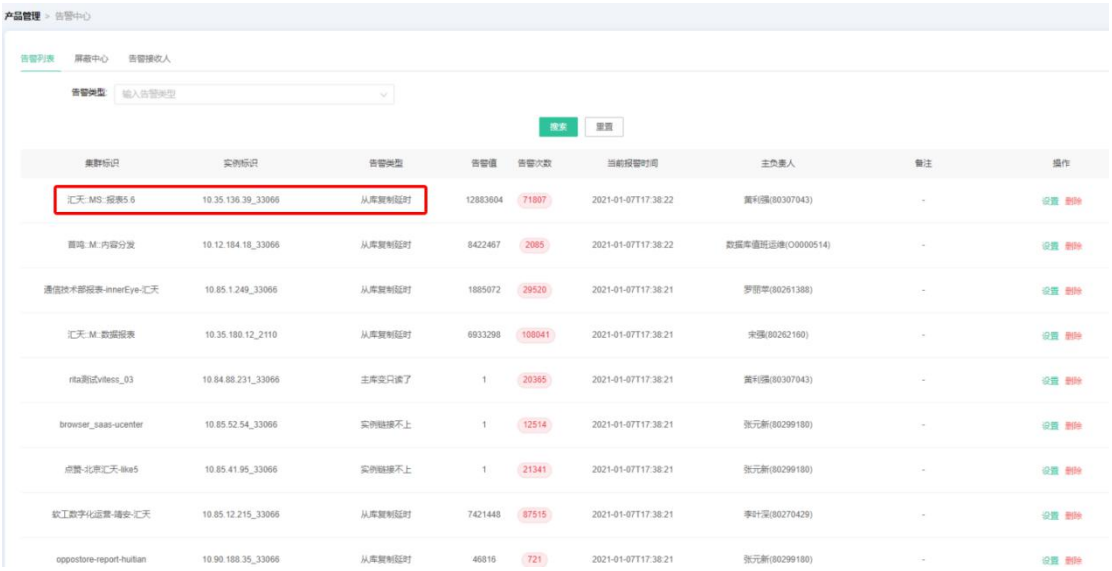
三、目前云平台 mysql 数据库面临的主要问题

3.1 目前云平台 mysql 数据库主要问题：

- 大表大批量 dml 操作，表上没有主键、索引，主从复制延迟，丢失高可用，无法迁移上云
- Load data 导致主从复制延迟，binlog 文件过大，写入性能低
- 大表查询、复杂查询、多表关联性能差
- 使用分区表的 open file 过大报错及全局 MDL 锁

3.2 PostgreSQL 以上业务场景对比测试

3.2.1 PG 物理流复制，大表 DML 操作从库无复制延时问题



```
mysql> show slave status \G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 10.35.136.38
                  Master_User: slave
                  Master_Port: 33066
                Connect_Retry: 60
              Master_Log_File: mysql-bin.001644
          Read_Master_Log_Pos: 147996543
               Relay_Log_File: relay-log.000085
                Relay_Log_Pos: 155871711
        Relay_Master_Log_File: mysql-bin.000020
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table: test.%,performance_schema.%,information_schema.%
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 155871548
              Relay_Log_Space: 2457732656793
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 12883791
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 103833066
                  Master_UUID: de0f3906-e0ce-11e9-9ccb-e4434b18db38
             Master_Info_File: mysql.slave_master_info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
      Slave_SQL_Running_State: Reading event from the relay log
           Master_Retry_Count: 86400
                  Master_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
               Master_SSL_Crl:
           Master_SSL_Crlpath:
           Retrieved_Gtid_Set:
            Executed_Gtid_Set:
                Auto_Position: 0
1 row in set (0.00 sec)
```

```
mysql>
mysql> show relaylog events in 'relay-log.000085' from 155871711 limit 100;
+------------------+-----------+-------------+-----------+-------------+----------------------------------------------+
| Log_name         | Pos       | Event_type  | Server_id | End_log_pos | Info                                         |
+------------------+-----------+-------------+-----------+-------------+----------------------------------------------+
| relay-log.000085 | 155871711 | Query       | 103833066 |   155871622 | BEGIN                                        |
| relay-log.000085 | 155871785 | Rows_query  | 103833066 |   155871676 | # delete from rpt_gh_version_dis             |
| relay-log.000085 | 155871839 | Table_map   | 103833066 |   155871752 | table_id: 198 (ieg_da.rpt_gh_version_dis)    |
| relay-log.000085 | 155871915 | Delete_rows | 103833066 |   155879947 | table_id: 198                                |
| relay-log.000085 | 155880110 | Delete_rows | 103833066 |   155888142 | table_id: 198                                |
| relay-log.000085 | 155888305 | Delete_rows | 103833066 |   155896337 | table_id: 198                                |
| relay-log.000085 | 155896500 | Delete_rows | 103833066 |   155904532 | table_id: 198                                |
| relay-log.000085 | 155904695 | Delete_rows | 103833066 |   155912727 | table_id: 198                                |
| relay-log.000085 | 155912890 | Delete_rows | 103833066 |   155920922 | table_id: 198                                |
| relay-log.000085 | 155921085 | Delete_rows | 103833066 |   155929117 | table_id: 198                                |
| relay-log.000085 | 155929280 | Delete_rows | 103833066 |   155937310 | table_id: 198                                |
| relay-log.000085 | 155937473 | Delete_rows | 103833066 |   155945492 | table_id: 198                                |
| relay-log.000085 | 155945655 | Delete_rows | 103833066 |   155953703 | table_id: 198                                |
| relay-log.000085 | 155953866 | Delete_rows | 103833066 |   155961901 | table_id: 198                                |
| relay-log.000085 | 155962064 | Delete_rows | 103833066 |   155970105 | table_id: 198                                |
| relay-log.000085 | 155970268 | Delete_rows | 103833066 |   155978319 | table_id: 198                                |
| relay-log.000085 | 155978482 | Delete_rows | 103833066 |   155986531 | table_id: 198                                |
| relay-log.000085 | 155986694 | Delete_rows | 103833066 |   155994695 | table_id: 198                                |
| relay-log.000085 | 155994858 | Delete_rows | 103833066 |   156002891 | table_id: 198                                |
| relay-log.000085 | 156003054 | Delete_rows | 103833066 |   156011056 | table_id: 198                                |
| relay-log.000085 | 156011219 | Delete_rows | 103833066 |   156019238 | table_id: 198                                |
| relay-log.000085 | 156019401 | Delete_rows | 103833066 |   156027438 | table_id: 198                                |
| relay-log.000085 | 156027601 | Delete_rows | 103833066 |   156035640 | table_id: 198                                |
| relay-log.000085 | 156035803 | Delete_rows | 103833066 |   156043812 | table_id: 198                                |
| relay-log.000085 | 156043975 | Delete_rows | 103833066 |   156052023 | table_id: 198                                |
| relay-log.000085 | 156052186 | Delete_rows | 103833066 |   156060192 | table_id: 198                                |
| relay-log.000085 | 156060355 | Delete_rows | 103833066 |   156068389 | table_id: 198                                |
| relay-log.000085 | 156068552 | Delete_rows | 103833066 |   156076587 | table_id: 198                                |
```

```
mysql> show create table rpt_gh_version_dis;
+--------------------+---------------------------------------------------------------
---------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------
| Table              | Create Table


+--------------------+---------------------------------------------------------------
---------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------
| rpt_gh_version_dis | CREATE TABLE `rpt_gh_version_dis` (
  `dt_month` date NOT NULL DEFAULT '0000-00-00' COMMENT '统计月份',
  `dt` varchar(32) NOT NULL DEFAULT '0000-00-00' COMMENT '统计日期',
  `model` varchar(32) DEFAULT '0' COMMENT '机型',
  `os_version_name` varchar(32) DEFAULT '0' COMMENT 'OS版本',
  `sdk_version_name` varchar(32) DEFAULT '0' COMMENT 'SDK版本',
  `users` int(11) DEFAULT '0' COMMENT '用户数'
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COMMENT='指标层_游戏大厅_版本分布报表'
/*!50100 PARTITION BY RANGE (TO_DAYS(dt_month))
(PARTITION pmin VALUES LESS THAN (735234) ENGINE = MyISAM,
 PARTITION pmax VALUES LESS THAN MAXVALUE ENGINE = MyISAM) */                |
+--------------------+---------------------------------------------------------------
---------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------
1 row in set (0.00 sec)

mysql>
```

该表 rpt_gh_version_di 有 8725852 行数据，无主键无索引，delete from rpt_gh_version_dis 造成主从复制延时，relaylog 从 2020-08-11 开始堆积，堆积 2.3T
PostgreSQL 下主库端删除大表主从复制测试，验证如下：

```
postgres=#
postgres=# \d+ pg_big_table
                                    Table "public.pg_big_table"
     Column      |          Type          | Collation | Nullable | Default | Storage  | Stats target | Description
-----------------+------------------------+-----------+----------+---------+----------+--------------+-------------
 l_orderkey      | integer                |           |          |         | plain    |              |
 l_partkey       | integer                |           |          |         | plain    |              |
 l_suppkey       | integer                |           |          |         | plain    |              |
 l_linenumber    | integer                |           |          |         | plain    |              |
 l_quantity      | numeric(15,2)          |           |          |         | main     |              |
 l_extendedprice | numeric(15,2)          |           |          |         | main     |              |
 l_discount      | numeric(15,2)          |           |          |         | main     |              |
 l_tax           | numeric(15,2)          |           |          |         | main     |              |
 l_returnflag    | character(1)           |           |          |         | extended |              |
 l_linestatus    | character(1)           |           |          |         | extended |              |
 l_shipdate      | date                   |           |          |         | plain    |              |
 l_commitdate    | date                   |           |          |         | plain    |              |
 l_receiptdate   | date                   |           |          |         | plain    |              |
 l_shipinstruct  | character(25)          |           |          |         | extended |              |
 l_shipmode      | character(10)          |           |          |         | extended |              |
 l_comment       | character varying(44)  |           |          |         | extended |              |
Access method: heap

postgres=# select count(*) from pg_big_table;
  count
----------
 59986052
(1 row)

Time: 2548.848 ms (00:02.549)
postgres=#
```

主库端删除大约 6000 万行的大表，删除完成后从库立即同步完成，PG 物理流复制，主从复制无延迟。

```
postgres=#
postgres=# delete from pg_big_table;
DELETE 59986052
Time: 51408.097 ms (00:51.408)
postgres=#
postgres=# \x
Expanded display is on.
postgres=# select * from pg_stat_replication ;
-[ RECORD 1 ]----+------------------------------
pid              | 571191
usesysid         | 34597
usename          | repl
application_name | walreceiver
client_addr      | 10.90.76.8
client_hostname  |
client_port      | 55060
backend_start    | 2021-01-11 16:45:00.594029+08
backend_xmin     | 173638209
state            | streaming
sent_lsn         | 22C/C4EA1598
write_lsn        | 22C/C4EA1598
flush_lsn        | 22C/C4EA1598
replay_lsn       | 22C/C4EA1598
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2021-01-11 17:44:17.558829+08

Time: 6.940 ms
postgres=#
```

3.2.2 PG 物理流复制，load data 从库无复制延时问题，load data 性能更高
导入 6000 万行数据，从库无复制延时

```
postgres=#
postgres=# Copy lineitem FROM '/home/tpch/2.18.0_rc2/dbgen/lineitem.csv' WITH DELIMITER AS '|';
COPY 59986052
Time: 174065.236 ms (02:54.065)
postgres=#
postgres=# \x
Expanded display is on.
postgres=# select * from pg_stat_replication ;
-[ RECORD 1 ]----+------------------------------
pid              | 571191
usesysid         | 34597
usename          | repl
application_name | walreceiver
client_addr      | 10.90.76.8
client_hostname  |
client_port      | 55060
backend_start    | 2021-01-11 16:45:00.594029+08
backend_xmin     | 173638231
state            | streaming
sent_lsn         | 22E/B4C1A898
write_lsn        | 22E/B4C1A898
flush_lsn        | 22E/B4C1A898
replay_lsn       | 22E/B4C1A898
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2021-01-11 18:24:03.604547+08

Time: 7.323 ms
postgres=#
postgres=#
```

MySQL & PG load data 性能对比

	MySQL 耗时	MySQL 备注	PG 耗时	PG 备注
6000 万	434.27 秒 Query OK, 59986052 rows affected (7 min 14.27 sec) Records: 59986052 Deleted: 0 Skipped: 0 Warnings: 0	Binlog 过大 主从复制延时	173.54 秒 COPY 59986052 Time: 173542.533 ms (02:53.543)	主从复制无延时

3.2.3 PG 在支持复杂业务场景时有更好的性能

集群名称: 汇天-贷超-后台报表

主要问题：该集群为典型的 OLAP 报表系统，频繁地 load data/存在大量的慢查询/主从复制延时明显。

对以下慢 SQL 迁移在 PG 上进行对比测试，性能得到明显提升。另外，PG load data 相比 MySQL 也有更好性能。

慢 SQL01:

```
alter table rpt_loan_channel_admission_convert_d truncate partition p20210110;
```

慢 SQL02:

```
SELECT db_loan_report.rpt_wallet_buscard_use_d.dayno AS col_29541_dayno, sum(use_cards)/sum(use_card_users) AS met_21688_direct_0
FROM db_loan_report.rpt_wallet_buscard_use_d
WHERE db_loan_report.rpt_wallet_buscard_use_d.dayno >= '2020-10-13 00:00:00' AND db_loan_report.rpt_wallet_buscard_use_d.dayno <= '2021-01-10 23:59:59' GROUP BY db_loan_report.rpt_wallet_buscard_use_d.dayno ORDER BY col_29541_dayno desc
 LIMIT 500000;
```

慢 SQL03:

```
SELECT count(*) AS count_1
FROM (SELECT db_loan_report.rpt_loan_mart_source_report_v2.dt AS col_32824_dt, sum(db_loan_report.rpt_loan_mart_source_report_v2.homepage_uv) AS met_24361_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.certification_uv) AS met_24362_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.verify_cnt) AS met_24363_direct_0, sum(verify_cnt)/sum(certification_uv) AS met_24446_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.credit_apply_dcnt) AS met_24365_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.credit_pass_dcnt) AS met_24366_direct_0, sum(credit_pass_cnt)/sum(credit_apply_cnt) AS met_24449_direct_0,
```

sum(credit_limit_sum)/sum(credit_pass_dcnt) AS met_24448_direct_0, sum(credit_rate_sum)/sum(credit_pass_dcnt) AS met_24447_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.loan_apply_dcnt) AS met_24372_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.loan_success_dcnt) AS met_24374_direct_0, sum(loan_success_cnt)/sum(loan_apply_cnt) AS met_24450_direct_0, sum(db_loan_report.rpt_loan_mart_source_report_v2.loan_success_amount) AS met_24377_direct_0
FROM db_loan_report.rpt_loan_mart_source_report_v2
WHERE db_loan_report.rpt_loan_mart_source_report_v2.dt >= '2020-01-01 00:00:00' AND db_loan_report.rpt_loan_mart_source_report_v2.dt <= '2020-12-31 00:00:00' AND db_loan_report.rpt_loan_mart_source_report_v2.first_channel_name_group IN ('unknown', 'other', '广告', '体系内合作', '软件商店', '浏览器', '短信', '游戏中心', 'OPPO 商城', '我的 OPPO')
GROUP BY db_loan_report.rpt_loan_mart_source_report_v2.dt) AS expr_qry;

SQL 语句编号	表数据量	MySQL 慢日志中记录时间(单位秒)	PG 库执行时间
SQL01	155590	Query_time: 584.204737 Lock_time: 372.631809	0.02 直接 truncate 对应分表或 DETACH truncate table rpt_loan_channel_admission_convert_d_p20210110;
SQL02	3827833	30.106177	Time: 67.457 ms 0.067s
SQL03	9309832	31.442047	Time: 71.208 ms 0.071s

四、PG 高可用方案

4.1 PostgreSQL 常用的高可用工具

	PATRONI	PAF	REPMGR
特点	支持 REST APIs 和 HAproxy 集成 支持回调脚本(callbacks scripts) DCS(Etcd/zookeeper/consul/kubernets)	Pacemaker and Corosync stack.	配置简单 不支持 DCS 会在数据库中创建 repmgr 系统表 自动切换需要配置节点 ssh 互信

4.2 Patroni

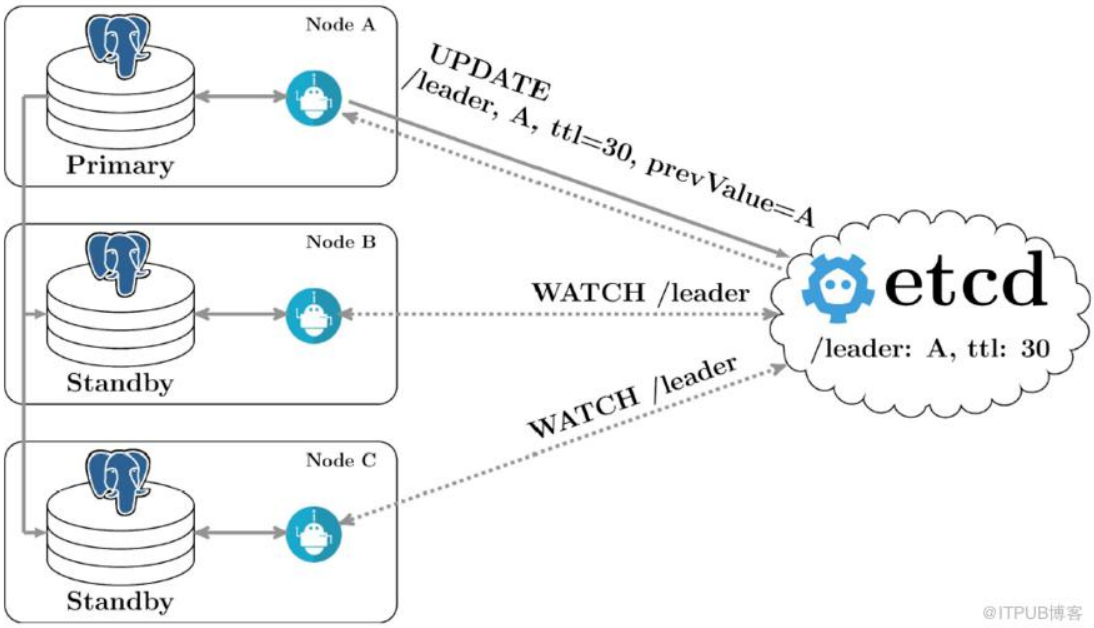
Patroni 是目前 PostgreSQL 数据库国内应用广泛、成熟的开源高可用工具。

- 由 Zalando 公司开发(Zalando 是总部位于德国柏林的大型网络电子商城,PG 重度用户)
- 100% Open source
- 支持自动、手动 switchover/failover
- 通过 DCS(etcd, zookeeper, Consul 等)保存集群元数据，DCS 自身高可用
- 支持配置 callbacks scripts
- 支持 Rest API
- 支持集成 HAproxy

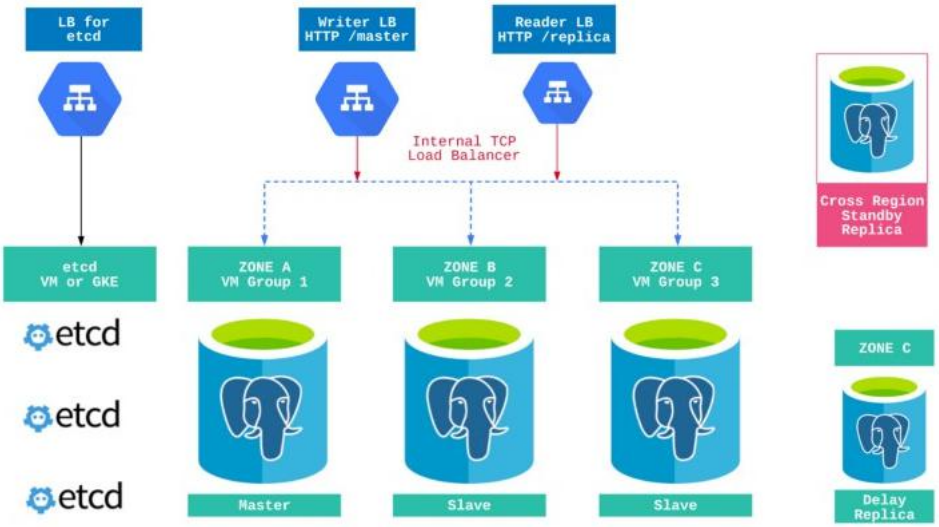
一主两从环境：

```
[postgres@localhost ~]$ patronictl list
+ Cluster: pgsql (6919000762281282249) --+---------+----+-----------+
| Member | Host                | Role    | State   | TL | Lag in MB |
+--------+---------------------+---------+---------+----+-----------+
| pg01   | 192.168.56.94:54322 | Leader  | running |  8 |           |
| pg02   | 192.168.56.95:54322 | Replica | running |  8 |         0 |
| pg03   | 192.168.56.96:54322 | Replica | running |  8 |         0 |
+--------+---------------------+---------+---------+----+-----------+
[postgres@localhost ~]$
```

Patroni 集群基本架构图：

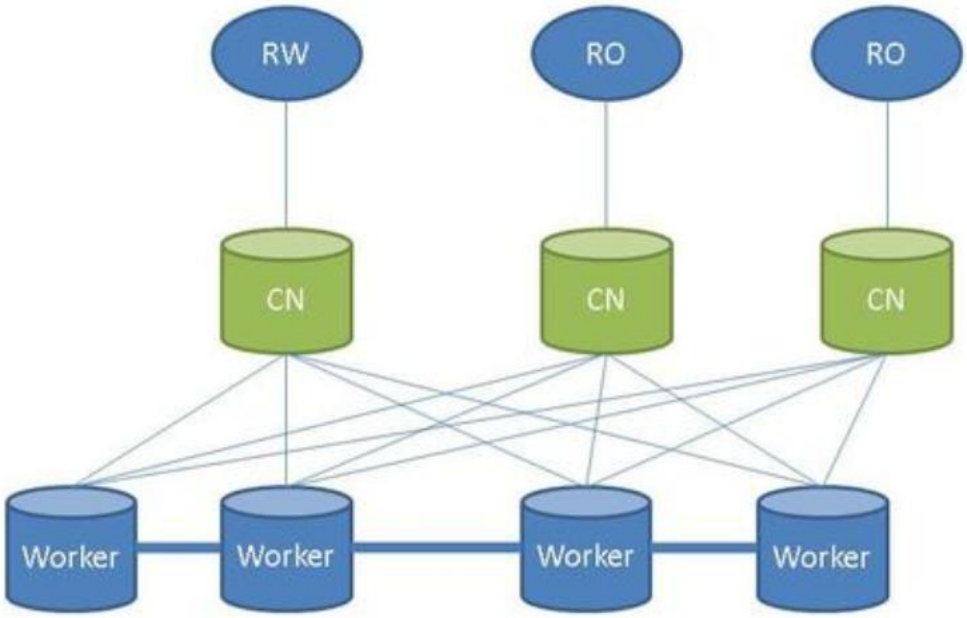


跨区高可用 Patroni 架构：



4.3 分布式插件 Citus

Citus 是开源的 PostgreSQL 分布式数据库插件，由协调器节点(coordinator)和 Worker 节点构成一个数据库集群，worker 节点可以作为分片分库分表，支持水平扩容。应用将查询发送到协调器节点，协调器处理后发送至各个 work 节点并行执行，并返回最终数据。
https://www.citusdata.com/



Citus 读写分离架构：