Tips de monitoreo con SQL

POSTGRESQL – MYSQL

INSTRUCTOR: EMANUEL CALVO

Premisas

CONCEPTOS BÁSICOS

Premisas

- Evitar/monitorear la latencia de escritura y lectura.
  - Es lo que tardan los dispositivos de almacenamiento, en devolver los datos.
- Aumentar/monitorear el rendimiento de salida de resultados.
  - Capacidad de respuesta del servidor.
- Seguridad.

Bases de datos veloces, implican aplicaciones Veloces.

Como lograr …?

- Menor latencia de I/O:
  - Más discos.
  - Discos más rápidos.
  - Tablespaces y particiones separadas entre si.
  - Utilización de sistemas de ficheros más avanzados.
- Rendimiento:
  - Más memoria.
  - Más y mejores procesadores y núcleos.
  - Mejor conexión entre servidores en red.
  - Servidores dedicados.

Monitoreo Básico

- Desde el Sistema Operativo.
  - Se obtiene el rendimiento en términos específicos del sistema operativo y hardware.
- Desde el motor de Base de datos.
  - Se puede obtener datos de accesos a objetos y cantidad de datos en caché de los mismos.

Monitoreo de Base

- Estado de los accesos a relaciones.
- Estado de las estadísticas.
- Estados de los índices.
- Estado del caché.
- Procesos en ejecución.
- TPS con pgbench, medir rendimiento.
- Consultas lentas.

Tamaño de la base y tablas

Postgresql	**Mysql**
• Select pg_size_pretty(pg_database_size(name)); • SELECT pg_size_pretty(pg_total_relation_size(tabla));	• SELECT table_schema "Data Base Name", sum(data_length + index_length) / 1024 / 1024 "Data Base Size in MB" FROM information_schema.TABLES GROUP BY table_schema ; • SELECT table_schema "Data Base Name", sum(data_length + index_length) / 1024 / 1024 "Data Base Size in MB", sum(data_free)/ 1024 / 1024 "Free Space in MB" FROM information_schema.TABLES GROUP BY table_schema ;

Tamaño tablas

Postgresql

- SELECT pg_size_pretty(pg_total_relation_size(tabla));
- SELECT pg_size_pretty(pg_relation_size(tabla));

Mysql

- **SELECT** table_name, table_rows, data_length, index_length, round(((data_length + index_length) / 1024 / 1024),2) "Size in MB" **FROM** information_schema.**TABLES WHERE** table_schema = "schema_name";

Procesos

Postgresql	**Mysql**
• SELECT * FROM pg_stat_activity;	• SHOW PROCESSLIST; • SHOW STATUS LIKE '%threads%'; • show session status like 'connections';

Mysql

APARTADO DE CONSULTAS PARA MYSQL

Engines (Mysql)

- Show engines;
- show engine innodb status;

InnoDB Status

```
Per second averages calculated from the last 59 seconds
-----------------BACKGROUND THREAD-----------------
srv_master_thread loops: 4 1_second, 0 sleeps, 0 10_second, 5 background, 5 flush
srv_master_thread log flush and writes: 0
----------SEMAPHORES----------
OS WAIT ARRAY INFO: reservation count 3, signal count 3
Mutex spin waits 1, rounds 30, OS waits 0
RW-shared spins 3, rounds 90, OS waits 3
RW-excl spins 0, rounds 0, OS waits 0
Spin rounds per wait: 30.00 mutex, 30.00 RW-shared, 0.00 RW-excl
------------TRANSACTIONS------------
Trx id counter 502
Purge done for trx's n:o < 32C undo n:o < 0
History list length 7
LIST OF TRANSACTIONS FOR EACH SESSION:
---TRANSACTION 0, not started, OS thread id 2988
MySQL thread id 6, query id 110 localhost 127.0.0.1 root
show engine innodb status
---TRANSACTION 501, not started, OS thread id 3568
MySQL thread id 4, query id 105 localhost 127.0.0.1 root
---TRANSACTION 0, not started, OS thread id 3552
MySQL thread id 2, query id 59 localhost 127.0.0.1 root
--------FILE I/O--------
I/O thread 0 state: wait Windows aio (insert buffer thread)
I/O thread 1 state: wait Windows aio (log thread)
I/O thread 2 state: wait Windows aio (read thread)
I/O thread 3 state: wait Windows aio (read thread)
I/O thread 4 state: wait Windows aio (read thread)
I/O thread 5 state: wait Windows aio (read thread)
I/O thread 6 state: wait Windows aio (write thread)
I/O thread 7 state: wait Windows aio (write thread)
I/O thread 8 state: wait Windows aio (write thread)
I/O thread 9 state: wait Windows aio (write thread)
Pending normal aio reads: 0 [0, 0, 0, 0] , aio writes: 0 [0, 0, 0, 0] ,
 ibuf aio reads: 0, log i/o's: 0, sync i/o's: 0
Pending flushes (fsync) log: 0; buffer pool: 0
191 OS file reads, 7 OS file writes, 7 OS fsyncs
0.00 reads/s, 0 avg bytes/read, 0.00 writes/s, 0.00 fsyncs/s
```

```
--------------------INSERT BUFFER AND ADAPTIVE HASH INDEX-------------------
Ibuf: size 1, free list len 0, seg size 2, 0 merges
merged operations:
 insert 0, delete mark 0, delete 0
discarded operations:
 insert 0, delete mark 0, delete 0
Hash table size 195193, node heap has 0 buffer(s)
0.00 hash searches/s, 0.00 non-hash searches/s
---LOG---
Log sequence number 3308699
Log flushed up to   3308699
Last checkpoint at  3308699
0 pending log writes, 0 pending chkp writes
10 log i/o's done, 0.00 log i/o's/second
---------BUFFER POOL AND MEMORY----------------------
Total memory allocated 49971200; in additional pool allocated 0
Dictionary memory allocated 18671
Buffer pool size   3008
Free buffers       2829
Database pages     179
Old database pages 0
Modified db pages  0
Pending reads 0
Pending writes: LRU 0, flush list 0, single page 0
Pages made young 0, not young 0
0.00 youngs/s, 0.00 non-youngs/s
Pages read 179, created 0, written 1
0.00 reads/s, 0.00 creates/s, 0.00 writes/s
No buffer pool page gets since the last printout
Pages read ahead 0.00/s, evicted without access 0.00/s
LRU len: 179, unzip_LRU len: 0
I/O sum[0]:cur[0], unzip sum[0]:cur[0]
----------ROW OPERATION------------
0 queries inside InnoDB, 0 queries in queue
1 read views open inside InnoDB
Main thread id 796, state: waiting for server activity
Number of rows inserted 0, updated 0, deleted 0, read 11
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
```

Explorando tablas

```
mysql> select * from
information_schema.tables
where TABLE_NAME like 'prueba' limit
1\G
******* 1. row *****************
TABLE_CATALOG: def
TABLE_SCHEMA: mysql
TABLE_NAME: prueba
TABLE_TYPE: BASE TABLE
ENGINE: InnoDB
VERSION: 10
ROW_FORMAT: Compact
TABLE_ROWS: 5
AVG_ROW_LENGTH: 3276
DATA_LENGTH: 16384
MAX_DATA_LENGTH: 0
INDEX_LENGTH: 0
DATA_FREE: 0
AUTO_INCREMENT: 6
CREATE_TIME: 2010-11-
03 12:09:45
UPDATE_TIME: NULL
CHECK_TIME: NULL
TABLE_COLLATION:
latin1_swedish_ci
CHECKSUM: NULL
CREATE_OPTIONS:
TABLE_COMMENT:
1 row in set (0.00 sec)
```

Explarando Tablas (2)

- show table status from mysql like 'prueba'\G

Particiones

```
mysql> select * from
information_schame.partitions where
table_name = 'prueba' limit 1\G
********** 1. row ***************************
TABLE_CATALOG: def
TABLE_SCHEMA: mysql
TABLE_NAME: prueba
PARTITION_NAME: NULL
SUBPARTITION_NAME: NULL
PARTITION_ORDINAL_POSITION: NULL
SUBPARTITION_ORDINAL_POSITION: NULL
PARTITION_METHOD: NULL
SUBPARTITION_METHOD: NULL
PARTITION_EXPRESSION: NULL
SUBPARTITION_EXPRESSION: NULL
PARTITION_DESCRIPTION: NULL
TABLE_ROWS: 5
AVG_ROW_LENGTH: 3276
DATA_LENGTH: 16384
MAX_DATA_LENGTH: NULL
INDEX_LENGTH: 0
DATA_FREE: 0
CREATE_TIME: 2010-11-03 12:09:45
UPDATE_TIME: NULL
CHECK_TIME: NULL
CHECKSUM: NULL
PARTITION_COMMENT:
NODEGROUP:
TABLESPACE_NAME: NULL
1 row in set (0.02 sec)
```

Estadísticas

```
mysql> select * from statistics
where table_name like 'prueba' limit 1\G
********* 1. row ***************************
TABLE_CATALOG: def
 TABLE_SCHEMA: mysql
   TABLE_NAME: prueba
   NON_UNIQUE: 0
 INDEX_SCHEMA: mysql
   INDEX_NAME: PRIMARY
 SEQ_IN_INDEX: 1
  COLUMN_NAME: the_key
    COLLATION: A
  CARDINALITY: 5
     SUB_PART: NULL
       PACKED: NULL
     NULLABLE:
   INDEX_TYPE: BTREE
      COMMENT:
INDEX_COMMENT:
1 row in set (0.02 sec)
```

Working example

```
mysql> show table status like 'prueba'\G
Name: prueba
        Engine: InnoDB
       Version: 10
    Row_format: Compact
          Rows: 9703
 Avg_row_length: 35
   Data_length: 344064
Max_data_length: 0
  Index_length: 0
     Data_free: 0
 Auto_increment: 16371
   Create_time: 2010-11-03 12:09:45
   Update_time: NULL
    Check_time: NULL
     Collation: latin1_swedish_ci
      Checksum: NULL
```

```
mysql> delete from prueba where a between
      80 and 81;  Query OK, 218 rows affected (0.30
      sec)
mysql> show table status like 'prueba'\G
Name: prueba
        Engine: InnoDB
       Version: 10
    Row_format: Compact
          Rows: 10823
 Avg_row_length: 31
   Data_length: 344064
Max_data_length: 0
  Index_length: 0
     Data_free: 0
 Auto_increment: 16371
   Create_time: 2010-11-03 12:09:45
   Update_time: NULL
    Check_time: NULL
     Collation: latin1_swedish_ci
      Checksum: NULL
```

Working example (2)

```
mysql> optimize table prueba\G
 1. row ****************************
  Table: mysql.prueba
     Op: optimize
Msg_type: note
Msg_text: Table does not support
    optimize, doing recreate + analyze
    instead
 2. row ****************************
  Table: mysql.prueba
     Op: optimize
Msg_type: status
Msg_text: OK
2 rows in set (2.13 sec)
```

```
mysql> show table status like 'prueba'\G
1. row ****************************
           Name: prueba
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 10508
 Avg_row_length: 31
    Data_length: 327680
Max_data_length: 0
   Index_length: 0
      Data_free: 0
 Auto_increment: 13300
    Create_time: 2010-11-03 12:09:45
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
```

Postgresql

APARTADO CONSULTAS PARA POSTGRESQL

Monitoreo de Base

- Estado de los accesos a relaciones.
  - Pg_statio_user_tables
  - Pg_stat_user_tables
  - Tamaños
    - select pg_size_pretty(pg_database_size(‘ejemplo'));
    - select pg_size_pretty(pg_relation_size('datos'::regclass));

Monitoreo de Base

- Estado de las estadísticas.
  - Pg_stats
  - SELECT * FROM pg_stats WHERE tablename = ‘tabla’ AND attname = ‘columna’;

Monitoreo de Base

- Estados de los índices.
  - Pg_stat_user_indexes
  - Pg_statio_user_indexes

Monitoreo de Base

- Estado del caché.
  - Cantidad de bloques leídos:
    - Select pg_stat_get_db_blocks_fetched((select datid from pg_stat_database where datname = 'pampabs'));
  - Bloques leídos y en caché:
    - Select pg_stat_get_db_blocks_hit((select datid from pg_stat_database where datname = 'pampabs'));

Consultas lentas

Postgresql

- Consultas lentas.
  - Activar log_min_duration_statement (milisegundos).

Mysql

- Consultas lentas
  - log_slow_queries
  - log_queries_not_using_indexes

Trucos

- Realizando un ALTER TABLE sin modificar la tabla se obliga la reestructuración.
- Ordenar los registros físicamente:
  - SELECT * INTO tabla2 FROM tabla ORDER BY columna;

Contribs Postgresql

ALGUNOS CONTRIBS PARA MONITOREO

pg_stat_statements

- Loguea absolutamente todas las consultas.
- Funciones:
  - pg_stat_statements_reset ()
- Se utiliza:
  - Select * from pg_stat_statements;

```
#Configuracion en el Postgresql.conf:

shared_preload_libraries='pg_stat_statements'
custom_variable_classes='pg_stat_statements'
pg_stat_statements.max = 10000
pg_stat_statements.track = all
pg_stat_statements.save = on
```

pgfouine

- Descargar desde *pgfoundry.org*.
- Necesita un php-cli.
- Si tenemos que usar stderr, el log_line_prefix debe establecerse en:
  - '%t [%p]: [%l-1]'

Php pgfouine.php -format html-with-graphics –logtype stderr –file <archivo_log> > result.html

pgstattuple

- Varias funciones de medición estadística.
- Solo hacen un bloqueo de lectura.

- pgstattuple(‘table’)
- pgstatindex(‘index’)
- pg_relpages

pgrowlocks

- Devuelve la información por tupla.
- Devuelve lock_type.

```
SELECT *
    FROM
            general g JOIN
            pgrowlocks(‘general’) p
            ON (g.ctid = p.locked_row);
```

Contacte a: postgres.arg (at) gmail.com

Skype: emanuel.cfranco

Gracias Por asistir