MYSQL维护指南

1. 监控数据库性能

1.1. 主机监控

定期监控和分析数据库的性能指标，包括查询速度、连接数、CPU和内存使用情况等。这有助于识别性能瓶颈，并采取相应的优化措施。

top: **free -h** **df -h** 检查mysql主机的性能，可用资源应该冗余30%以上。

1.2. 状态监控

```
-- 这条命令用于查询当前有多少客户端线程连接到数据库。
-- 这个数字可以帮助你了解数据库的负载情况，如果连接数异常高，可能意味着你的数据库正面临过高的请求负载或者连接泄露问题
SHOW GLOBAL STATUS LIKE 'Threads_connected';

-- 这条命令用于显示自数据库启动以来执行的查询总数。
-- 这个指标可以帮助你评估数据库的工作量，了解数据库的使用频率
SHOW GLOBAL STATUS LIKE 'Queries';

-- 这条命令查询排序缓冲区的大小。
-- 排序缓冲区是MySQL用于排序操作的内存区域。调整这个值可以影响排序操作的性能，特别是在处理大量数据时
SHOW VARIABLES LIKE 'sort_buffer_size';

-- 这条命令用于查询InnoDB缓冲池的大小。InnoDB缓冲池是存储InnoDB表数据和索引的内存区域。
-- 这个值的大小直接影响到数据库的性能，特别是在处理大型数据库时
SHOW VARIABLES LIKE 'innodb_buffer_pool_size';

-- 这条命令显示当前MySQL服务器上所有运行的进程信息。
-- 这可以帮助你识别当前正在执行的查询，检查是否有长时间运行的查询或者锁竞争情况
SHOW FULL PROCESSLIST;
```

1.3. 利用**MySQL**的性能模式（**Performance Schema**）

```
-- 检查performance_schema这个系统变量的状态，如果performance_schema=ON，说明performance_schema已经启用
SHOW VARIABLES LIKE 'performance_schema';

-- 如果没有启用performance_schema，可以通过修改my.cnf文件来启用performance_schema，添加如下配置
-- [mysqld]
-- performance_schema=ON
-- 所有已执行SQL语句的性能摘要，返回了执行次数最多的前20个查询模板，这有助于识别出数据库中最频繁执行的查询，这些查询往往是性能优化的重点目标
use performance_schema;

SELECT
    SCHEMA_NAME,
    DIGEST_TEXT,
    COUNT_STAR as `执行次数`,
    SUM_TIMER_WAIT / COUNT_STAR / 1e12 AS `平均执行时间（秒）`,
    MAX_TIMER_WAIT / 1e12 as `最大执行时间（秒）`,
    SUM_LOCK_TIME / COUNT_STAR / 1e12 AS `平均锁定时间（秒）`,
    SUM_ROWS_SENT / COUNT_STAR AS `平均返回的行数`,
    SUM_ROWS_EXAMINED / COUNT_STAR AS `平均扫描的行数`,
    SUM_ROWS_AFFECTED / COUNT_STAR AS `平均影响的行数`,
    SUM_CREATED_TMP_DISK_TABLES / COUNT_STAR AS `平均创建的临时磁盘表`,
    SUM_SORT_MERGE_PASSES / COUNT_STAR AS `序操作的效率`,
    SUM_NO_INDEX_USED / COUNT_STAR AS `没有使用索引的次数`,
    SUM_NO_GOOD_INDEX_USED / COUNT_STAR AS `没有更好的使用索引的次数`
FROM
    events_statements_summary_by_digest
WHERE
    SCHEMA_NAME NOT IN (
        'mysql',
        'performance_schema',
        'information_schema'
    )
ORDER BY
    `平均执行时间（秒）` DESC
LIMIT
    20;
```

1.4. 分析慢日志

分析查询日志和慢查询日志，找出执行时间长、效率低下的查询语句，并进行优化。

# 开启慢查询日志
slow_query_log=1

慢查询日志文件路径
slow_query_log_file=/var/log/mysql/mysql-slow.log
如果未设置，可以使用 SHOW VARIABLES LIKE 'slow_query_log_file'查看

# 记录查询执行时间超过30秒的查询
long_query_time=10

- - 查询是否开启慢日志
SHOW VARIABLES LIKE 'slow_query_log';

- - 临时开启慢日志，重启后失效
SET GLOBAL slow_query_log = 'ON';

- - 查看慢查询阈值
SHOW VARIABLES LIKE 'long_query_time';

- - 设置超过阈值的查询时间为慢查询
SET GLOBAL long_query_time = 10;

- - 慢日志存储路径
SHOW VARIABLES LIKE 'slow_query_log_file';

- - 查看慢查询日志
SELECT

*

FROM

mysql.slow_log
ORDER BY

start_time DESC
LIMIT
20;

2. 备份数据库

根据数据库的重要性和数据变动频率，制定合理的备份策略。通常建议每天或每周进行一次全备份，并结合增量备份以减少存储空间和时间消耗。

2.1. 备份方式
可以选择逻辑备份（如使用mysqldump工具）或物理备份（如直接复制数据文件）。逻辑备份生成SQL脚本文件，便于在不同版本的MySQL之间迁移；物理备份则速度更快，恢复时更接近原始状态。

- - 备份mysql数据库
mysqldump - u root - p --all-databases > all_databases.sql

- - 恢复mysql数据库
mysql - u root - p < all_databases.sql

2.2. 验证备份

定期验证备份数据的完整性和可恢复性，确保在需要时能够成功恢复数据。

3. 优化查询语句和表结构

3.1. 优化查询语句
使用合适的索引、避免全表扫描、减少不必要的JOIN操作等，以提高查询效率。

3.2. 优化表结构
定期检查并优化表结构，删除不必要的字段和表，合并相似的表以减少冗余数据。对于大型表，可以使用OPTIMIZE TABLE命令进行优化，以减少磁盘空间使用和提高查询性能。

3.3. 归档历史数据

将超过一定历史时限的数据库进行归档，并从实时数据库进行删除，以降低数据量提高数据库性能。

4. 更新和维护

4.1. 更新数据库软件：
定期更新MySQL服务器和相关组件的软件版本，以修复已知的漏洞、提高性能和增加新功能。

4.2. 安装安全补丁：
及时安装MySQL的安全补丁，以防止黑客攻击和数据泄露。

5. 安全管理

5.1. 设置强密码与密码密码过期策略

为用户和数据库设置复杂且不易猜测的密码，并定期更换密码。设置密码过期策略

- - 全局的密码生命周期
SET GLOBAL default_password_lifetime = 180;

- - 设置密码过期时间
ALTER USER 'your_username' @'localhost' PASSWORD EXPIRE INTERVAL 180 DAY;

5.2. 最小权限原则：

遵循最小权限原则：为用户和应用程序分配必要的最小权限集合，以减少安全风险。
定期审计：定期审计数据库的访问日志和权限分配情况，及时发现并处理潜在的安全问题。
不允许使用ROOT直接：不允许使用ROOT用户直连数据库，应该为每个应用程序单独设置连接用户。

6. 其他维护任务

6.1. 检查磁盘空间

确保数据库服务器有足够的磁盘空间来存储数据和日志文件。

6.2. 清理无用数据

定期清理无用的数据、临时表和日志文件等，以释放磁盘空间和提高性能。

6.3. 配置和优化MySQL参数
根据监控结果和实际需求，调整MySQL的配置参数，如缓冲区大小、连接数等，以优化数据库性能。

版本号 #6
由 董列涛 创建于 16 七月 2024 06:16:38
由 董列涛 更新于 2 四月 2026 08:06:31