河北师范大学软件学院
Software College of Hebei Normal University

——高性能PHP应用开发之

第5讲 MySQL数据库优化

目录

CONTENTS

目录

CONTENTS

数据库优化

- 数据库优化是我们在日常开发和工作面试中一定会遇到的一个问题。
- 数据库优化一种综合的技术，它并不是通过一种方式就能把MySQL的性能提高多少，而是优化一点好一点，最后总体性能得到提升。
- 对于一个以数据为中心的应用，数据库的好坏直接影响到程序的性能，因此数据库性能至关重要。要保证数据库的效率，可以从以下几方面入手。

① 选择何种类型数据库?　② 数据库表结构设计

③ 选择合适的数据存储引擎　④ 定位慢查询，添加索引

⑤ 数据缓存　⑥ 复制及读写分离　⑦ 垂直切分和水平切分

① 选择何种类型数据库?

- 数据库类型：
  - ✓ 关系型数据库：mysql/oracle/db2/informix/sysbase/sql server等等。
  - ✓ 非关系型数据库（NoSQL，Not Only SQL）：MongoDB等。
- 如何选择?
  - ✓ NoSQL天生适合分布式系统，而且大多针对某种类型的数据、某种使用场景做过优化；比如大批量的监控数据，用MySQL存储费时费力，可以选择NoSQL；另一方面，时间序列数据库，使用NoSQL存取会有数量级提升。
  - ✓ 关系型数据库：传统数据应用领域，数据稳定的场合，网站开发大多数情况均使用此种形式。

② 数据库表结构设计

- 关系型数据库，数据表之间的关系是否符合范式?

✓ 三范式设计：原子性、唯一性、无冗余。

✓ 反范式设计：没有冗余的数据库未必是最好的数据库，有时为了提高运行效率，就必须降低范式标准，适当保留冗余数据。

在表的1对N的情况下，为了提高效率，可能会在1的这张表中设计冗余字段。

③ 选择合适的存储引擎

以MySQL数据库为例进行探讨，MySQL数据库支持多种形式的存储引擎，最常用的主要有以下两种。

- **MyISAM**：MyISAM是MySQL的默认存储引擎。其能够快速查询唯一键数据，支持全文索引，磁盘空间占用较少；但是不支持事务机制，不支持外键关联。
- **InnoDB**：InnoDB 符合ACID原则的存储引擎，支持事务，可以放弃和回滚查询，支持多种联机备份策略，提高了应用程序在高负载、大量连接下的并发能力；但是不支持全文索引，简单查询速度可能低于MyISAM。
- 选择何种存储引擎?

✓ 当数据库95%操作是读取数据操作时，应该选用MyISAR存储引擎。

✓ 当事务性和一致性非常重要时，当包含多表复杂模式时，当不间断操作非常重要时，当多用户并发情况经常出现时，应该选择InnoDB存储引擎。

④ 定位慢查询，添加索引

- 定位慢查询：MySQL中定位慢查询，主要通过慢查询日志来定位；打开MySQL的慢查询日志很简单，只需修改MySQL配置文件，重新启动MySQL服务器即可。
  - ✓ slow-query-log-file：指定慢查询日志的文件名
  - ✓ long_query_time：定义超过3秒的查询即为慢查询
- 执行 show variables like ‘%slow%’；查看慢查询日志是否开启。
- 执行SQL语句后，慢查询会自动记录到慢查询日志文件中。
- 执行 show status like ‘%slow%’；查询当前存在多少条慢查询。

```
log-slow-queries=mysql_slow.log
long_query_time=3
```

```
mysql> show variables like '%slow%';
+---------------------+----------------+
| Variable_name       | Value          |
+---------------------+----------------+
| log_slow_queries    | ON             |
| slow_launch_time    | 2              |
| slow_query_log      | ON             |
| slow_query_log_file | mysql_slow.log |
+---------------------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> show status like '%slow%';
+---------------------+-------+
| Variable_name       | Value |
+---------------------+-------+
| Slow_launch_threads | 0     |
| Slow_queries        | 2     |
+---------------------+-------+
2 rows in set (0.00 sec)
```

④ 定位慢查询，添加索引

- 分析慢查询：使用EXPLAIN关键字，可以了解MySQL正在进行什么样的查询操作，这可以帮助我们发现瓶颈的所在，并显示出查询或表结构在哪里出了问题。使用方法很简单，只需要在 SQL语句之前加上 explain 关键字即可。

```
mysql> explain select article_title from articles order by article_title limit 10;
+----+-------------+----------+------+---------------+------+---------+------+--------+----------------+
| id | select_type | table    | type | possible_keys | key  | key_len | ref  | rows   | Extra          |
+----+-------------+----------+------+---------------+------+---------+------+--------+----------------+
|  1 | SIMPLE      | articles | ALL  | NULL          | NULL | NULL    | NULL | 123675 | Using filesort |
+----+-------------+----------+------+---------------+------+---------+------+--------+----------------+
```

④ 定位慢查询，添加索引

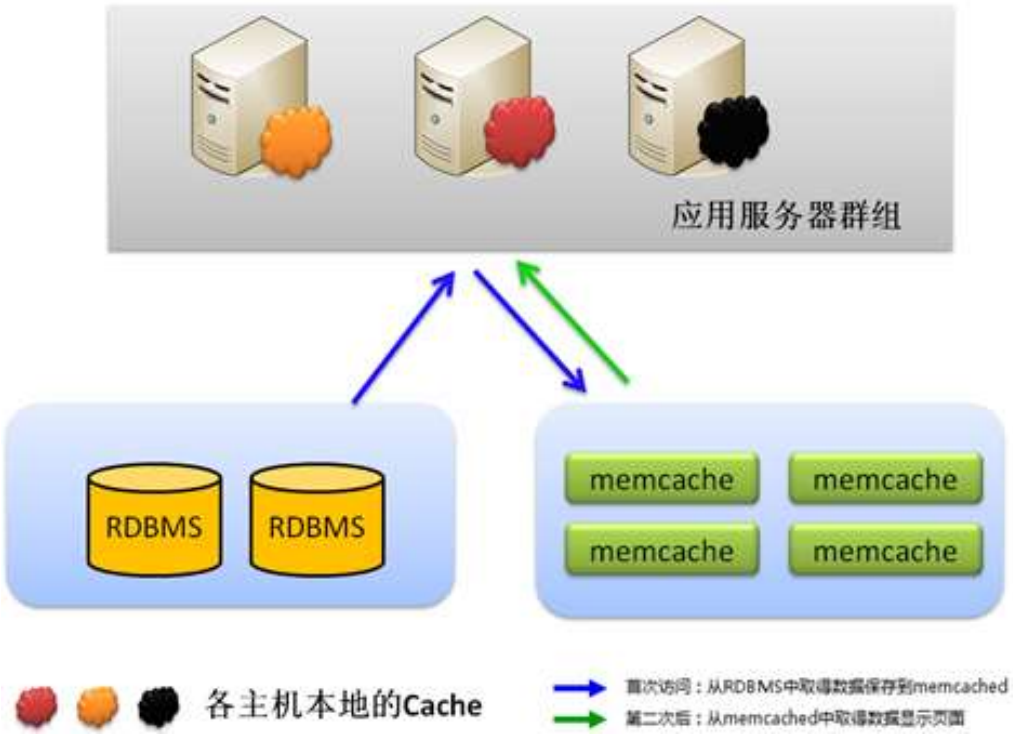
- 添加索引：对于提高数据库性能，索引是最物美价廉的东西了。不用加内存，不用改程序，不用调SQL，只要执行正确的‘create index’，查询速度就可能提高百倍千倍。

```
CREATE INDEX title_idx on articles (article_title);
```

```
mysql> explain select article_title from articles order by article_title limit 10;
+----+-------------+----------+-------+---------------+-----------+---------+------+------+-------------+
| id | select_type | table    | type  | possible_keys |    key    | key_len | ref  | rows | Extra       |
+----+-------------+----------+-------+---------------+-----------+---------+------+------+-------------+
|  1 | SIMPLE      | articles | index | NULL          | title_idx | 767     | NULL | 10   | Using index |
+----+-------------+----------+-------+---------------+-----------+---------+------+------+-------------+
```

⑤ 数据缓存

常见的数据库，比如oracle、mysql等，数据都是存放在磁盘中。虽然在数据库层也做了对应的缓存，但这种数据库层次的缓存一般针对的是查询内容，而且粒度也太小，一般只有表中数据没有变更的时候，数据库对应的cache才发挥作用。但这并不能减少业务系统对数据库产生的增、删、查、改的庞大IO压力。所以数据库缓存技术在此诞生，实现热点数据的高速缓存，提高应用的响应速度，极大缓解后端数据库的压力。



⑥ 读写分离

开始就创建两个数据库连接是一个好的方法，一个用于读取，一个用于写入，并且允许不同数据库服务器连接它们。

在进行应用程序编码时，可以把更改数据的任何查询（UPDATE、INSERT、DELETE等）都写成使用写入连接，纯SELECT或读取查询则一律使用读取连接。

通过使用两个连接，可以轻松重新配置你的应用程序以支持大量不同的扩展选项，使用一个或多个从属服务器来增加查询带宽。

⑦ 数据库切分

数据库切分包括垂直切分和水平切分，实现方式上又包括分库、分表。垂直切分保证业务的独立性，防止不同业务争抢资源，毕竟业务是有优先级的；水平切分主要用于突破单机瓶颈。

切分后也可对不同片数据进行不同优化。如按时间切分，超过一定时间数据不允许修改，就可以引入压缩了，数据传输及读取减少很多。

垂直切分一般都要做，只不过业务粒度大小而已。就算当前压力小，也尽量分出几个逻辑库出来，等规模上去了，很方便迁移扩展。

水平拆分有一定难度，但如果将来一定会到这个规模，又可能用到，建议越早做越好。因为对应用的改动较大，而且迁移成本高。

其他注意事项

① 尽量避免 select ＊ 语句，使用哪些字段，就只查询这些字段。

② 永远为每张表设置一个ID，且其为自动增加类型。

③ 利用LIMIT 1取得唯一行。

④ 若能使用 ENUM ， 则不要使用 VARCHAR。

⑤ 使用 Join连接 代替 子查询。

⑥ ……

河北师范大学软件学院
Software College of Hebei Normal University

目录

CONTENTS

MySQL配置文件

- MySQL配置文件：my.ini ，位于 MySQL根目录下
  - ✓ MySQL启动后，将会自动读取配置文件中的配置项，从而应用相应配置
  - ✓ 若修改 my.ini 文件内容，需要重新启动 MySQL服务器以适应改变

- 配置文件结构：
  - ✓ 客户端部分： [client] 、[mysql]
  - ✓ 服务器端部分：[mysqld]

目录

CONTENTS

MariaDB

- MariaDB数据库管理系统是MySQL的一个分支，主要由开源社区在维护，采用GPL授权许可。 MariaDB的目的是完全兼容MySQL，包括API和命令行，使之能轻松成为MySQL的代替品。
- 在存储引擎方面，使用XtraDB来代替MySQL的InnoDB。 MariaDB由MySQL的创始人Michael Widenius主导开发，他早前曾以10亿美元的价格，将自己创建的公司MySQL AB卖给了SUN，此后，随着SUN被甲骨文收购，MySQL的所有权也落入Oracle的手中。
- MariaDB名称来自Michael Widenius的女儿Maria的名字。

MariaDB与MySQL相比较

- 与 MySQL 相比较，MariaDB 更强的地方在于：
- 更快的复制查询处理
- 更少的警告和bug
- 运行速度更快
- 更好的功能测试
- 慢查询日志的扩展统计
- ……

PostgreSQL

- PostgreSQL是以加州大学伯克利分校计算机系开发的 POSTGRES，现在已经更名为PostgreSQL，版本 4.2为基础的对象关系型数据库管理系统（ORDBMS）。PostgreSQL支持大部分 SQL标准并且提供了许多其他现代特性：**复杂查询、外键、触发器、视图、事务完整性、MVCC**。同样，PostgreSQL可以用许多方法扩展，比如， 通过增加新的数据类型、函数、操作符、聚集函数、索引。免费使用、修改、和分发 PostgreSQL，不管是私用、商用、还是学术研究使用。

PostgreSQL优点

- 它包括了可以说是目前世界上最丰富的数据类型的支持，其中有些数据类型可以说连商业数据库都不具备， 比如 IP 类型和几何类型等。
- PostgreSQL 是全功能的自由软件数据库，很长时间以来，PostgreSQL 是唯一支持事务、子查询、多版本并行控制系统（MVCC）、数据完整性检查等特性的唯一的一种自由软件的数据库管理系统。
- PostgreSQL拥有一支非常活跃的开发队伍，而且在许多黑客的努力下，PostgreSQL 的质量日益提高。

PostgreSQL缺点

- 过于学院味，因为首先它的目的是数据库研究，因此不论在稳定性， 性能还是使用方方面面，长期以来一直没有得到重视。
- 欠缺一些比较高端的数据库管理系统需要的特性，比如数据库集群，更优良的管理工具和更加自动化的系统优化功能 等提高数据库性能的机制等。

河北师范大学软件学院
Software College of Hebei Normal University

感谢聆听！

THANK YOU FOR YOUR ATTENTION