DOI:10.19850/j.cnki.2096-4706.2021.06.021

一种 Python ORM 框架性能测试分析方法研究

贺宗平，贺曦冉，秦新国

（南京审计大学 信息化办公室，江苏 南京 211815）

摘 要：在使用 Python 进行系统应用开发的过程中，对各类型的关系型数据库，如 MySQL、Oracle、SQL Server 等的增、删、改、查操作，需要基于 ORM 框架规范的概念和模型。因此从安全性、可扩展性等角度分析测试 Python 社区生态中的多种 ORM 框架，以及适用于不同的应用场景和条件，并提出对 ORM 框架性能进行横向对比测试分析的方法和实例，具有较高的实用价值。

关键词：Python；ORM；数据库；性能测试

中图分类号：TP392　　**文献标识码**：A　　**文章编号**：2096-4706（2021）06-0083-04

Research on a Test and Analysis Method of Python ORM Framework Performance

HE Zongping，HE Xiran，QIN Xinguo

（Information Office，Nanjing Audit University，Nanjing 211815，China）

Abstract：In the process of carrying out system application development using Python，the URUD operations on various types of relational database，such as MySQL，Oracle，SQL Server and so on，needs to base on the concept and model of ORM framework specification. Therefore，it is of great practical value to analyze and test many ORM frameworks in Python community ecology from the angles of security and scalability，as well as applying to different application scenarios and conditions，and put forward methods and examples of horizontal comparative test and analysis of ORM framework performance.

Keywords：Python；ORM；database；performance testing

0 引 言

在利用 Python 进行 Web 应用系统进行如网络数据可视化分析、知识图谱应用相关的开发过程中，其中涉及大量对各类型数据库的访问操作，如 MySQL、Oracle、SQL Server 等，主要是“增、删、改、查”（CRUD）。在应用程序开发中，从访问性能、安全性和可扩展性等方面考虑，对数据库相关访问操作往往是通过 ORM 框架来实现。Python 社区生态中有多种 ORM 框架，适用于不同的应用场景和条件，因此对 ORM 框架性能进行横向基准测试分析，在不同的场景需求下选择适合的 ORM 框架，对于提升应用系统性能具有重要的价值意义。

1 ORM 概述

1.1 基本概念

在没有 ORM 框架条件下，如果系统程序需要操作数据库，需要在开发中编写原生 SQL 语句，并通过数据驱动接口模块远程操作数据库，如图 1 所示。

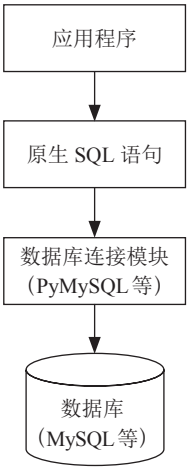

图 1 数据库直连操作访问示意图

这个数据库操作过程可以概括为以下几个步骤：

（1）建立数据库连接，获得连接对象。

（2）根据用户的操作定义组装 SQL 执行语句。

（3）用连接对象执行 SQL 语句，获得结果集数据对象。

（4）最后释放连接资源，关闭连接对象。

这种操作流程逻辑复杂、重复度高，并存在注入攻击等安全风险问题，系统的设计也无法做到业务与数据逻辑分离，不利于降低系统模块间的紧耦合关系。在系统开发中编写原生 SQL 语句会存在数据库适配迁移问题，例如针对 Oracle 开发的 SQL 语句无法平移应用到其他数据库上，一

收稿日期：2021-02-12

基金项目：江苏高校哲学社会科学研究项目（2020SJA0354）；江苏省高等学校教育技术研究会高校教育信息化研究课题（2019JSETKT060）；南京审计大学 2019 年度高教所课题（2019JG045）；南京审计大学 2020 年度高教所课题（2020JG051）

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

旦需要迁移改变数据库类型，会在源代码层级带来巨大的变更成本。

1.2 ORM 作用

为解决 Python 系统开发数据库操作的问题，实践中引入了 ORM 框架的概念。ORM（Object Relational Mapping）即对象关系映射，通过对各种类型数据库驱动连接库进行封装，避免对数据库直接编写 SQL 的操作，如图 2 所示。

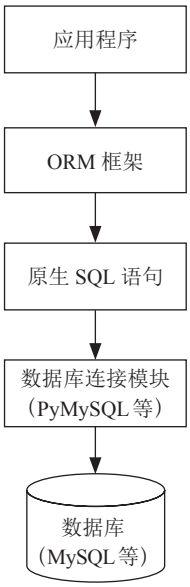

图 2 数据库 ORM 操作访问示意图

主要作用表现在以下几个方面：

（1）统一访问接口。通用数据库交互接口是 ORM 设计表现核心，使得能够以统一的方式与各种不同类型的数据库进行访问交互，避免了各种不同数据库 SQL 语法不同的问题。

（2）简化访问方式。ORM 封装提供了完整的数据库操作细节流程，不再需要开发人员构造烦琐的数据库访问流程。

（3）实现分层模式。ORM 解决了面向对象设计与关系型数据库的匹配问题，建立了对象与关系之间的映射框架，是 MVC 分层模式中的内存对象持久化存储具体实现方式。

（4）提高安全特性。由于 ORM 框架无须用户自定义拼接 SQL 语法等操作，从而阻断了外部的注入攻击等威胁，提升代码的安全性。

1.3 ORM 模型

ORM 框架中的类对应简称为模型，模型是数据的结构性定义，包含存储数据的字段类型和方法。每个模型都对应映射到某个数据表，模型中的属性对应数据表的字段。以 Django ORM 为例进行 CRUD 操作示例：

（1）在 ORM 框架中操作数据，必须先创建模型对应数据库表。创建示例模型：

```
class Employee():
id=models.AutoField(primary_key=True)
name=models.CharField(max_length=16)
gender=models.BooleanField(default=1)
birth=models.DateField()
department=models.CharField(max_length=30)
salary=models.DecimalField(max_digits=10,decimal_places=1)
```

（2）增加记录：

```
obj=Employee(name="John",gender=0,birth='1982-01-27',department=" 信息部 ",salary=1024)
obj.save()
```

（3）查询记录：查询所有名字为 John 的记录并获取第一条。

```
obj=Employee.objects.filter(name="John").first()
print(obj.id,obj.name,obj.birth)
```

（4）修改记录：过滤所有名字为 John 的记录并将 name 字段更新为 JOHN。

```
Employee.objects.filter(name="John").update(name="JOHN")
```

（5）删除记录：过滤出所有名字为 EGON 的记录并删除。

```
Employee.objects.filter(name="JOHN").delete()
```

2 Python ORM 典型框架

2.1 Django ORM

Django ORM 具备所有 ORM 框架的重要特性，允许以接近 SQL 的方式与数据库交互，操作简单、简洁并且开放。在 Django 中 model 代表数据信息的来源和结构，包含了存储数据的重要字段和行为 。通常用一个模型（model）映射到某个数据表，建立模型与数据存储之间的关联映射。Django ORM 框架概括起来包括三个方面：

（1）每个模型都是一个 Python 类，它是 django.db.models.Model 的子类。

（2）模型的每个属性都代表一个数据库字段。

（3）Django 为用户提供了一个自动生成的数据库访问 API。

2.2 Peewee

Peewee 是一个轻量级的 Python ORM，操作使用直观。Peewee 中一个 model 类代表一个数据库的表，一个 Field 字段代表数据库中的一个字段，而一个 model 类实例化对象则代表数据库中的一行。Peewee 使用主要包括两部分：

（1）定义 model。在使用的时候，根据需求先定义好 Model，然后可以通过 create_tables() 创建表，若是已经创建好数据库表了，可以通过 pwiz 脚本工具直接创建 model。其中，CharField、DateField、BooleanField 等这些类型与数据库中的数据类型一一对应。

（2）操作数据库。Peewee 中的数据模型实例可以直接进行增、删、改和查的操作。save() 对应增加数据；使用 delete().where().execute() 进行删除，where() 是条件、execute() 负责执行语句；使用 update().where() 进行更新数据；查询使用 select().where()，select() 是查询，where() 是条件，get() 是获取第 1 条数据。

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2.3 Pony ORM

Pony ORM 提供便捷化的查询语法，实现了自动查询优化，提供在线的 ORM ER 实体关系图编辑器。与 Django ORM 相比，Pony ORM 提供：标识映射模式、自动事务管理、自动缓存查询和对象、完全支持复合键。Pony ORM 支持全自动缓存、读取速度快，缺点是不支持批量插入。

2.4 SQLAlchemy ORM

SQLAlchemy ORM 是 Python 中一个使用较为普及的框架库，具备高性能的数据库访问设计，实现了完整的企业级持久模型。SQLAlchemy 设计将 SQL 数据库的量级和性能放在首位，其次是对象集合的抽象。因此 SQLAlchemy 不同于其他 PythonORM 框架所采用的 Active Record 模型，其采用了近似 JavaHibernate 的数据映射模型。在 SQLAlchemy 中，“增删改查”操作通过 DBSession 对象来创建完成。

2.5 Tortoise ORM

与 Python 中其他成熟 ORM 相比，Tortoise ORM 是一种异步 ORM 框架，基于 Python 中的 asyncio 异步标准库实现，支持原生的异步编程。在 IO 密集型的应用场景中，异步处理方式能提极大地升效率，从而弥补 Python 运算性能方面的短板。主要特点包括：

（1）开发测试便捷。测试框架使用现有的 Python Unittest 框架，只需要调用 initializer() 和 finalizer() 来设置和删除测试数据库。

（2）可组合支持 Django 模型。

（3）支持多种标准字段。

（4）支持复合查询 API。

3 性能测试方法

3.1 基本条件

ORM 性能测试的首先要考虑以下几个基本条件：

（1）ORM 框架至少支持 2 个以上的数据，例如 MySQL、SQLite。

（2）框架运行环境为 Python 3.7。

（3）框架处于积极更新维护中。

（4）能够生成指定模型的初始 DDL。

（5）处理一对多关系。

3.2 测试基准

3.2.1 测试内容

测试内容包括几个方面内容：

（1）插入：单条插入（IS），批量插入（IBH），批量导入（IBK）。

（2）过滤：大量数据过滤（FL），少量数据过滤（FS），字典过滤（FD），元组过滤（FT）。

（3）更新：整个结构数据更新（UW），部分字段数据更新（UP）。

（4）删除：按过滤条件删除（DL）。

（5）查询：按过滤条件查询（GS）。

3.2.2 测试数据表

测试操作的数据表包括三种：无关联关系小型表、有关联关系小型表、大型表：

（1）无关联关系小型表（small_table_nr）如表 1 所示。

表 1 无关联关系小型表

字段	类型
id	整数型自增主键
timestamp	日期时间类型
level	整数枚举型
text	可变字符类型

（2）有关联关系小型表（small_table_wr）如表 2 所示。

表 2 有关联关系小型表

字段	类型	关联关系
id	整数型自增主键	——
timestamp	日期时间类型	——
level	整数枚举型	——
text	可变字符类型	——
parent	整数型	large_table 外键一对一
child	整数型	large_table 一对多
knows	——	large_table 多对多

（3）大型表（large_table）如表 3 所示。

表 3 大型表

字段	类型
id	整数型自增主键
created_at	日期时间类型
updated_at	日期时间类型
level	整数枚举型
text	可变字符类型
col_float	单精度浮点型
col_smallint	短整型
col_int	整型
col_bigint	长整型
col_char	可变字符型
col_text	长文本型
col_decimal	双精度浮点型
col_json	Json 型

3.3 测试结果

3.3.1 SQLite 测试结果

SQLite 测试数据结果如表 4 所示，Peewee 和 Pony ORM 有显著的性能提升，SQLAlchemy 和 Django ORM 性能表现近似，Tortoise ORM 读取速度慢，创建、更新和删除操作速度较快。

3.3.2 MySQL 测试结果

MySQL 测试数据结果如表 5 所示，Peewee 和 Tortoise ORM 有较好的性能表现，Pony ORM 性能表现相对更快，SQLAlchemy 和 Django ORM 操作速度则表现相对更慢。

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

表 4　SQLite 测试结果对比表　　单位：rows per second

测试 1	Django ORM	Peewee	Pony ORM	SQLAlchemy	Tortoise ORM
IS	3 708.87	5 988.94	5 313.19	1 787.23	8 771.86
IBH	5 202.03	7 080.62	20 682.10	9 936.82	11 856.23
IBK	22 641.95	34 830.97	—	36 263.39	92 215.74
FL	70 827.06	40 292.41	186 303.67	80 556.02	192 864.71
FS	16 590.77	16 877.80	11 548.63	16 673.05	24 988.74
GS	2 779.47	3 388.39	9 462.00	2 539.36	4 594.26
FD	95 383.31	59 175.90	106 239.43	84 593.08	350 127.46
FT	108 427.53	59 130.16	182 755.44	315 451.73	274 481.55
UW	3 393.08	5 845.30	22 510.83	18 488.72	12 511.45
UP	3 858.61	7 641.23	33 949.08	27 403.46	15 675.48
DL	4 053.31	11 102.10	49 233.60	51 217.01	21 317.97
均值	12 057.43	14 488.47	33 214.84	23 999.02	34 608.93

表 5　MySQL 测试结果对比表　　单位：rows per second

测试 2	Django ORM	Peewee	Pony ORM	SQLAlchemy	Tortoise ORM
IS	1 383.62	1 266.45	1 434.83	901.20	3 080.33
IBH	2 043.80	2 001.30	4 995.87	3 203.36	5 002.17
IBK	15 038.21	14 595.01	—	14 492.72	25 766.89
FL	54 917.59	26 408.01	159 944.32	47 911.41	31 345.78
FS	2 036.85	1 865.41	1 956.58	1 929.00	3 345.87
GS	1 181.04	1 003.87	4 031.27	1 103.81	1 654.70
FD	88 357.33	32 639.37	74 397.80	44 173.61	35 968.88
FT	96 285.53	32 741.59	160 698.59	103 112.61	33 153.43
UW	1 263.76	1 586.54	3 442.20	3 895.25	4 518.65
UP	1 464.38	2 032.87	5 431.92	6 084.50	6 197.01
DL	1 509.27	2 516.67	4 844.72	5 806.70	6 296.95
均值	5 478.73	4 514.32	9 979.18	7 195.1	8 389.33

4　结　论

通过测试数据结果分析，总体来说 Pony ORM 具有高度优化的性能表现，Django ORM 和 SQLAlchemy 在性能上表现上相似相近。Tortoise ORM 由于采用了异步框架支持，避免了 IO 读写上的等待时间耗费，具有十分出色的性能表现。Python 异步框架对于提升数据库相关操作效率具有十分重要的意义，融合异步技术将是 Python ORM 框架发展重点方向。

参考文献：

[1] 杨立苑，李芬，周雪莹，等 . 面向气象 Web 应用的数据库访问性能优化及应用 [J]. 计算机与数字工程，2020，48（11）：2671−2676.

[2] 熊学锋，彭小庆，曹鑫 . 基于改进 ORM 的 Oracle 数据库异构资源整合方法研究 [J]. 电子设计工程，2020，28（21）：38−41+46.

[3] 蹇常林 .ORM 在 Django 操作数据库中的应用 [J]. 技术与市场，2020，27（1）：56−57.

[4] 郭显娥 .Django 实现 ORM 模型数据查询优化 [J]. 山西大同大学学报（自然科学版），2019，35（3）：27−31+36.

[5] 熊伟，欧阳逸，张凌云 . 一种数据库访问代码自动生成方法 [J]. 广州大学学报（自然科学版），2019，18（3）：93−95.

[6] 陈忠菊 . 基于 SQLAlchemy 的研究和在数据库编程中的应用 [J]. 电脑编程技巧与维护，2015（1）：62+85.

[7] 郎芳 . 基于 Django 技术的自动化测试工具设计与实现 [D]. 西安：西安电子科技大学，2012.

作者简介：贺宗平（1982.09—），男，汉族，江苏南京人，工程师，硕士，主要研究方向：软件体系架构、数据平台。

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net