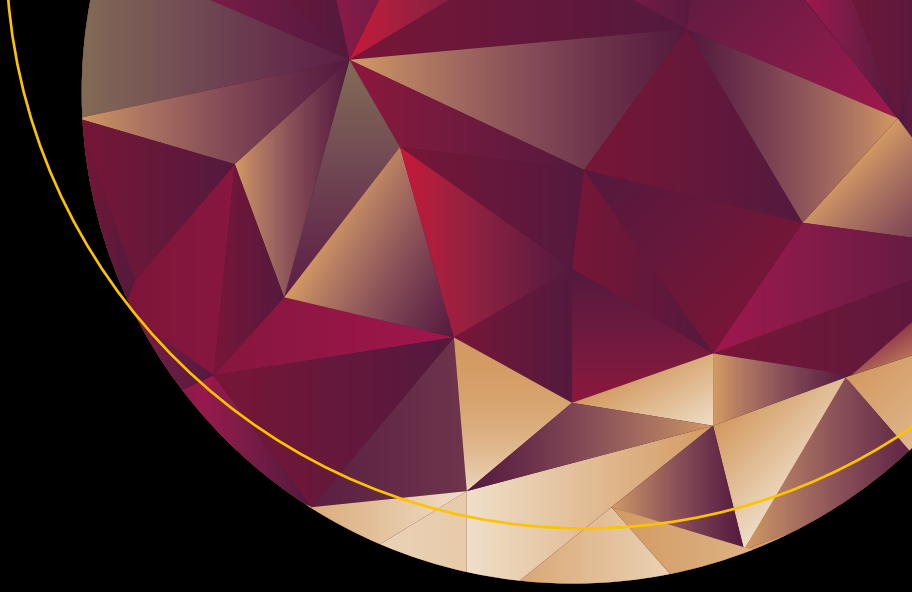

# MySQL Connector/Python Revealed

SQL and NoSQL Data Storage Using MySQL for Python Programmers

Jesper Wisborg Krogh

Apress®

# MySQL Connector/Python Revealed

## SQL and NoSQL Data Storage Using MySQL for Python Programmers

**Jesper Wisborg Krogh**

Apress®

*MySQL Connector/Python Revealed*

Jesper Wisborg Krogh
Hornsby, New South Wales, Australia

ISBN-13 (pbk): 978-1-4842-3693-2 ISBN-13 (electronic): 978-1-4842-3694-9
https://doi.org/10.1007/978-1-4842-3694-9

Library of Congress Control Number: 2018952522

Copyright © 2018 by Jesper Wisborg Krogh

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com/rights-permissions.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484236932. For more detailed information, please visit www.apress.com/source-code.

Printed on acid-free paper

*To my wife, Ann-Margrete, and my parents.*

# Table of Contents

# About the Author

**Jesper Wisborg Krogh** is a member of the Oracle MySQL Support team and has spoken on several occasions at Oracle OpenWorld. He has a Ph.D. in Computational Chemistry but he switched to working with MySQL and other software development in 2006. His areas of expertise include MySQL Cluster, MySQL Enterprise Backup, and the Performance and sys schemas. He is an active author in the Oracle Knowledge Base and regularly blogs on MySQL topics.

Jesper lives in Sydney, Australia, and enjoys spending time outdoors, walking, traveling, and reading.

# About the Technical Reviewer

**Charles Bell** conducts research in emerging technologies. He is a member of the Oracle MySQL Development team and works on various teams including Replication, Utilities, and MySQL Enterprise Backup. He received his Ph.D. in Engineering from Virginia Commonwealth University in 2005.

Charles is an expert in the database field and has extensive knowledge and experience in software development and systems engineering. His research interests include 3D printers, microcontrollers, 3D printing, database systems, software engineering, and sensor networks.

Charles lives in a small town in rural Virginia with his loving wife. He spends his limited free time as a practicing Maker, focusing on microcontroller projects and the refinement of 3D printers.

# Acknowledgments

I would like to thank all of the people who made this book possible. The Apress team has again been a great help, and I would in particular like to thank Jonathan Gennick, Jill Balzano, and Laura Berendson, the three editors I worked with while getting this book ready for production.

Several people have been invaluable sparring partners in technical discussions. Thanks to Charles Bell for providing a thorough and speedy technical review; his comments were, as always, very useful. The discussions with Nuno Mariz, Israel Gomez Delgado, and Philip Olson have also been invaluable.

Last but not least, thanks to my wife Ann-Margrete for her patience and support while I wrote this book.

# Introduction

MySQL Connector/Python is the official driver used by Python programs to communicate with MySQL. It is maintained by Oracle and is part of the MySQL suite of products. It allows you to connect to a MySQL database from your Python program and execute queries. You have a choice of several APIs including the option of using SQL statements or a NoSQL interface.

This book goes through the high-level usage as well as the low-level details. When you have read all ten chapters of this book, you will be able to decide which API is the best for your project and you'll be able to use it to execute your MySQL queries, handle errors, and troubleshoot when things go wrong.

## Book Audience

The book was written for developers who need to use MySQL as a backend data store or are otherwise interested in learning about the capabilities of MySQL Connector/Python and how to use it. No prior knowledge of MySQL Connector/Python is required. It is, however, an advantage to be familiar with databases in general and with SQL and Python.

## Book Structure

The chapters are divided into four parts. The journey starts out with some general background information, the installation of MySQL Connector/Python and MySQL Server, and the preparation for the example programs

that are included throughout the book. The next two parts are dedicated to each of the main APIs included in MySQL Connector/Python: the legacy (classic) API and the new X DevAPI. The final part discusses how to handle errors and how to troubleshoot.

## Part I - Getting Ready

The first part consists of just one chapter:

1. Introduction and Installation - This chapter starts out with an introduction to MySQL Connector/Python. It then goes through the process of downloading and installing it. The chapter finishes off with instructions for getting a MySQL Server instance set up for the example programs that are included throughout the rest of the book.

## Part II - The Legacy APIs

The second part goes through the details of the MySQL Connector/Python APIs based on the Python Database API specification (PEP 249), which is the API traditionally used when connecting from Python to MySQL. The four chapters are as follows:

2. Connecting to MySQL – The first task when using MySQL Connector/Python is to connect to the database instance. This chapter covers how to create and configure the connection. This chapter also includes a discussion of character sets and collations.

3. Basic Query Execution – In this chapter, you start executing queries. The discussion is split over two chapters; this chapter covers the basic parts including simple queries returning a single result set, using cursors, and the important topic of handling user input.

4. Advanced Query Execution – This chapter continues where the previous one ended. It goes through more advanced concepts of executing queries such as handling multiple result sets and loading data from a CSV file. The topic of connection properties is also revisited with a focus on the options that affect how transactions work, the behavior of queries, etc. Then there is a discussion of utilities, for example to test whether the connection is still alive. Finally, there is a discussion of the C Extension.

5. Connection Pooling and Failover – MySQL Connector/Python has built-in support for connection pooling and failover. This chapter discusses how to set up and use a connection pool as well as how to fail over to a different MySQL instance should the current become unavailable.

## Part III - The X DevAPI

The third part switches the focus to the new API called the X DevAPI. This API has reached general available status with MySQL 8.0 and provides uniform access from several programming languages. It includes support

for both NoSQL and SQL access as well as native support for working with JSON documents in the MySQL Document Store. The three chapters are as follows:

6. The X DevAPI – This chapter introduces the X DevAPI, including details of the parts that are shared between using MySQL as a document store and using SQL tables. It covers how to create connections and how to work with schemas, statements, and results.
7. The MySQL Document Store – While MySQL traditionally has been a relational SQL database, the MySQL Document Store allows you to use it to store JSON documents. This chapter goes through the details of how to use the X DevAPI to work with collections and documents.
8. SQL Tables – The X DevAPI also supports using MySQL with SQL tables, both using a NoSQL interface and to execute SQL statements. This chapter explains how to use the X DevAPI with SQL tables.

## Part IV - Error Handling and Troubleshooting

The fourth and final part covers two important topics: error handling and troubleshooting. The two chapters are as follows:

9. Error Handling – An important part of writing a program is knowing how to handle errors appropriately. This chapter covers errors from a MySQL Server and MySQL Connector/Python perspective including MySQL error numbers, SQL states, lock issues, and what to do when an error occurs.

10. Troubleshooting – When writing a program, something inevitably goes wrong. An error may occur or a query may not return the expected result. This chapter discusses how to find information that can help determine what the issue is and offers several examples of problems and their solution.

## Downloading the Code

The code for the examples shown in this book is available on the Apress web site, www.apress.com. A link can be found on the book's information page at https://www.apress.com/gp/book/9781484236932.

# PART I

# Getting Ready

CHAPTER 1

# Introduction and Installation

You are about to embark on a journey through the world of MySQL Connector/Python. Welcome aboard! This is the first chapter out of a ten-step guide that will take you through everything from installation to troubleshooting. Along the way you will become acquainted with the features and workings of the connector and its APIs.

This chapter will introduce MySQL Connector/Python by going through the versions, editions, and the APIs. The middle part of this chapter will discuss how to download and install the connector, and the final part will talk about MySQL Server, how to set up the server for the examples in this book, and a word on the examples themselves.

## Introduction

MySQL Connector/Python is the glue that is used between a Python program and a MySQL Server database. It can be used to manipulate the database objects using data definition language (DDL) statements as well as to change or query the data through data manipulation language (DML) statements.

You can also call MySQL Connector/Python a database driver. It is the official MySQL connector for Python, developed and maintained by Oracle Corporation by the MySQL development team. It effectively supports three different APIs, although only two are commonly used directly.

© Jesper Wisborg Krogh 2018
J. W. Krogh, *MySQL Connector/Python Revealed*,
https://doi.org/10.1007/978-1-4842-3694-9_1

This section introduces the MySQL Connector/Python versions, editions, and the three APIs.

## Versions

Before 2012, there was no Python connector maintained by Oracle. There were other third-party connectors, such as the MySQL-python (MySQLdb) interface; however, it was getting aged and only officially supported up to MySQL 5.5 and Python 2.7.

MySQL decided to develop its own connector: MySQL Connector/Python. It was written to be compatible with the MySQL-python interface and to be up to date with the latest MySQL Server and Python versions. The initial general availability (GA) release was version 1.0.7, which was released in September 2012. A major update occurred with version 2.1; it introduced the C Extension, which allows better performance. The latest GA release as of April 2018 is version 8.0.11, which additionally introduces the X DevAPI. This is the version that is the primary focus of this book.

---

**Note** If you look at the change history of MySQL Connector/Python, you may be a little puzzled. The version series before 8.0 was 2.1 with a few pre-GA releases of version 2.2. The list of 8.0 releases is no less puzzling: the latest pre-GA release is 8.0.6 with the first GA release being 8.0.11. Why the jumps? The version numbers of most MySQL products were aligned, which required some irregularity in release numbers, but it now means that MySQL Server 8.0.11 and MySQL Connector/Python 8.0.11 are released together.

---

It is recommended to use the latest patch release of the latest series of GA quality. Only the latest GA series receives all improvements and bug fixes. That means that, at the time of writing, it is recommended to use the latest MySQL Connector/Python 8.0 release. While the MySQL

Connector/Python 8.0 releases are coupled together with the release of MySQL Server and other MySQL products,[1] they are backward compatible with older MySQL Server versions. So, even if you are still using, for example, MySQL Server 5.7, you should still use MySQL Connector/Python 8.0.

---

**Tip** Use the latest release of the latest release series of GA quality to ensure you have access not only to all the latest features but also the latest available bug fixes. The latest MySQL Connector/Python version can be used with older MySQL Server versions. On the other hand, an older version of MySQL Connector/Python may not be compatible with the latest MySQL Server version. For example, MySQL Server 8.0 uses the `caching_sha2_password` authentication plugin by default, which is not supported until recently in MySQL Connector/Python.

---

As with any product under active development, new features are regularly added and bugs are fixed. You can follow the changes in the release notes, which are available from https://dev.mysql.com/doc/relnotes/connector-python/en/.

In addition to the various versions of MySQL Connector/Python, there are (as with other MySQL products) two different editions to choose from. Let's take a look at them.

## Community and Enterprise Editions

MySQL products are available in two different editions: Community and Enterprise. The Enterprise Edition is a commercial offering from Oracle. The difference between the two editions varies among the products. For example, for MySQL Server, several additional plugins are available for

[1] https://mysqlrelease.com/2018/03/mysql-8-0-it-goes-to-11/

the Enterprise Edition. For MySQL Connector/Python, the difference is subtler.

A common difference for all products is the license. The Community Edition is released under the GNU General Public License, version 2.0, whereas the Enterprise Edition uses a proprietary license. Additionally, the Enterprise Edition includes technical support through MySQL Technical Support Services. These are presently the only differences between the two editions for MySQL Connector/Python itself.

This book will work with either of the two editions and, except when briefly discussing download locations and install methods later in this chapter, there will be no mention of the edition. All examples have been written and tested with the Community Edition.

In contrast, when it comes to APIs, it makes a big difference which API you use.

## APIs

There are effectively three different APIs that can be used in MySQL Connector/Python. How to use the APIs is the main purpose of Chapters 2-9. Before you get started for real, it is worth taking a brief view of the differences.

Table 1-1 shows the three APIs, which MySQL Connector/Python module they are available in, the first GA version including support for the API, and the chapters where they are discussed.

***Table 1-1.*** *MySQL Connector/Python APIs*

| API | Module | First Version | Chapters |
|---|---|---|---|
| Connector/Python API | `mysql.connector` | 1.0.7 | 2, 3, 4, 5, 9, 10 |
| C Extension API | `_mysql_connector` | 2.1.3 | 4 |
| X DevAPI | `mysqlx` | 8.0.11 | 6, 7, 8, 9, 10 |

Additionally, the Connector/Python API and X DevAPI exist both in a pure Python implementation and one using C Extension under the hood. These two implementations are meant to be interchangeable. Some differences between the two implementations will be mentioned when encountered throughout the book.

As you can see, the main focus is on the Connector/Python API and X DevAPI. The Connector/Python API and the C Extension API exclusively use SQL statements to execute queries. The X DevAPI, on the other hand, supports NoSQL methods to handle JSON documents and SQL tables as well as support for SQL statements. The X DevAPI is a common API available for other programming languages as well, including JavaScript (Node.js), PHP, Java, DotNet, and C++.

So which API should you choose? From the description thus far, it sounds like it is a no-brainer to choose the X DevAPI. However, there is a little more to it than that.

If you are exclusively using SQL statements to execute queries, the C Extension and C Extension API are more mature. For example, they offer much better support for features such as parameter binding and prepared statements. If you need a connection pool, they are also the APIs to choose. If you have existing Python programs, they are also most likely using the Connector/Python API (with or without the C Extension implementation enabled).

On the other hand, the X DevAPI is a new API that has been designed from the ground up to fit modern requirements. The API also exists for other programming languages, making it easier to switch between languages when several languages are required for the applications. The NoSQL parts of the API makes simple queries against SQL tables and working with JSON documents simpler. The new command-line client, MySQL Shell, also supports using the X DevAPI via either Python or JavaScript. So, the X DevAPI there is a lot talking for new projects.

Since the X DevAPI is essentially in its version 1.0 (MySQL 8.0 is the first GA version for the X DevAPI), new features are more likely to become available in relatively short succession. If you are missing a feature, keep an eye on the release notes to see if the feature has become available, or register your interest at https://bugs.mysql.com/.

Whether to use the C Extension or not is to a large degree a question of performance compared to "convenience." The C Extension implementation provides better performance particularly when working with large result sets and prepared statements. However, the pure Python implementation is available on more platforms, is easier to work with when building MySQL Connector/Python yourself, and is easier to modify (as the name suggest, the pure Python implementation is written entirely in Python).

This concludes the introduction to MySQL Connector/Python. It is time to get started with the installation process. The first step is to download MySQL Connector/Python.

# Downloading

It is straightforward to download MySQL Connector/Python; however, there are still a few considerations. These considerations and the steps to perform the download are the topics of this section.

The first thing to ask is whether you need the Community or Enterprise Edition of the connector. This decides both the download and the install options. The Community Edition is available from several locations and both in the form of source code and as binary distributions. The Enterprise Edition is only available as the binary distribution from Oracle.

**Tip** The recommended way to install the Community Edition of MySQL Connector/Python is to use packages from the Python Packaging Authority (PyPa)/Python Package Index (PyPi). This is done using the `pip` tool and does not require predownloading any files. One downside of using PyPi is there can be a small lag from when the release is made to when it becomes available in PyPi.

Table 1-2 shows an overview of the delivery methods available for MySQL Connector/Python and whether the method is available for the Community and Enterprise Editions.

***Table 1-2.*** *MySQL Connector/Python Download Options*

| Distribution | Community Edition | Enterprise Edition |
|---|---|---|
| Python Packages (`pip`) | Available; see installation | |
| Windows Installer | Available | Available |
| MSI Installer | Available | Available |
| APT repository | Available | |
| SUSE repository | Available | |
| Yum repository | Available | |
| RPM downloads | Available | Available |
| DEB packages | Available | Available |
| Solaris package | Available | Available |
| macOS | Available | Available |
| Platform-independent tar or zip files | Available | Available |

As you can see, MySQL Connector/Python is available for a large range of platforms and in different distributions. The Community Edition is available directly using the `pip` command-line tool; using the MySQL Yum repository for Red Hat Enterprise Linux, Oracle Linux, and Fedora Linux; the MySQL APT repository for Debian and Ubuntu; and using the MySQL SUSE repository for SLES. The `pip` and package repository options are only available for the Community Edition.

---

**Tip** Both a MySQL Installer and a MSI Installer for MySQL Connector/Python are available for Microsoft Windows. If you want to use one of these installers, MySQL Installer is recommended because it also supports most of the other MySQL products.

---

Table 1-3 shows the URLs for the download locations for the various sources and installers. The MySQL repositories count as installers in this context even though they are more like definition files used with an installer.

***Table 1-3.*** *Download Sources*

| Source/Installer | URL |
|---|---|
| **Community:** | |
| MySQL Installer for Microsoft Windows | https://dev.mysql.com/downloads/installer/ |
| APT repository | https://dev.mysql.com/downloads/repo/apt/ |
| SUSE repository | https://dev.mysql.com/downloads/repo/suse/ |
| Yum repository | https://dev.mysql.com/downloads/repo/yum/ |
| MySQL downloads | https://dev.mysql.com/downloads/connector/python/ |
| GitHub | https://github.com/mysql/mysql-connector-python |
| **Enterprise:** | |
| My Oracle Support | https://support.oracle.com/ |
| Oracle Software Delivery Cloud | https://edelivery.oracle.com/ |

The Community Edition-related downloads are available from pages under https://dev.mysql.com/downloads. If you need the source code, it is available from the MySQL Downloads site and MySQL's GitHub repository.[2]

The Enterprise Edition is available either from the *Patches & Updates* tab in My Oracle Support (MOS) or from the Oracle Software Delivery Cloud (requires creating an account and signing in). MySQL customers are

[2]If you are familiar with git, it is recommended to use GitHub to get the source code because it allows you to easily switch branches, which includes older releases. However, as with PyPi, there can be a lag of the newest changes to be uploaded.