

# Database Programming

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

### 1 Databases

a library manager  
relational design of databases  
query, commit, rollback

### 2 MySQL

an open source database  
running MySQL: database creation  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

MCS 275 Lecture 24  
Programming Tools and File Management  
Jan Vershelde, 8 March 2010

# Database Programming

## Databases

**a library manager**  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

### 1 Databases

**a library manager**

relational design of databases  
query, commit, rollback

### 2 MySQL

an open source database  
running MySQL: database creation  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

# File Oriented Information Systems

a library manager

## Databases

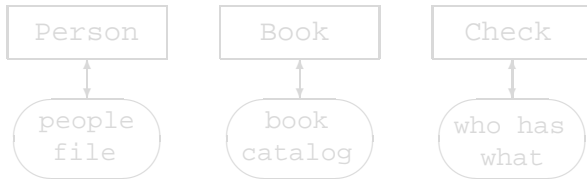
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

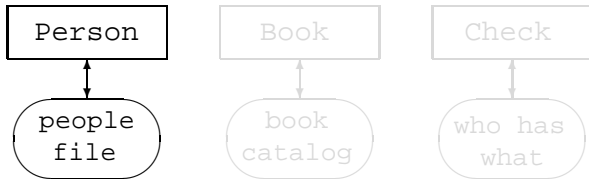
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

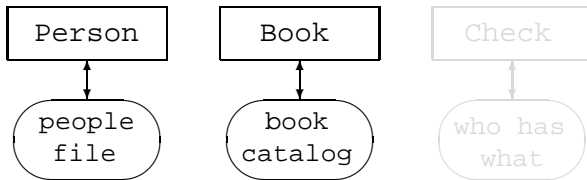
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

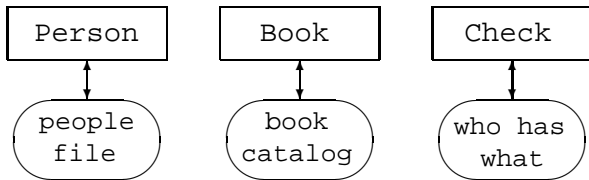
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

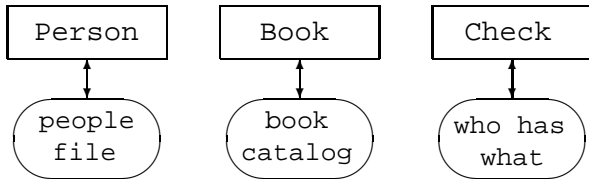
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

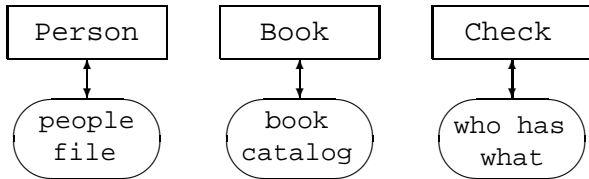
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# File Oriented Information Systems

a library manager

## Databases

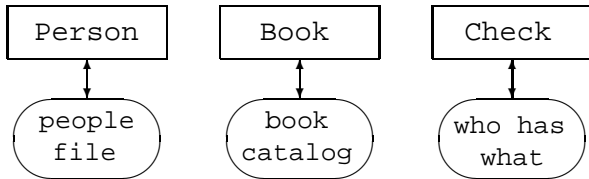
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

Consider the management of a library, using files:



The information system consists of 3 files:

- 1 records of people (librarians and patrons);
- 2 data catalog of books in the library;
- 3 who has checked out what books.

# Database Information Systems

## Databases

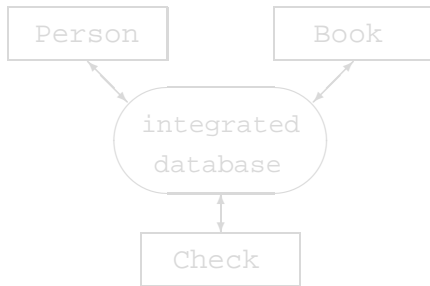
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A database is a collection of data organized and managed by a specific software system: the *Database Management System (DBMS)*.



To design databases we consider the *relational model*.

# Database Information Systems

## Databases

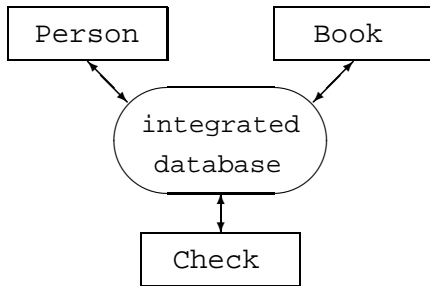
a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A database is a collection of data organized and managed by a specific software system: the *Database Management System (DBMS)*.



To design databases we consider the *relational model*.

# Database Programming

## Databases

a library manager  
**relational design of  
databases**  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

### 1 Databases

a library manager  
**relational design of databases**  
query, commit, rollback

### 2 MySQL

an open source database  
running MySQL: database creation  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

# The Relational Model

some terminology

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A *relation* is a table with a fixed number of columns.

The columns are called *attributes*, taking values belonging to a *domain* (similar to types).

The rows are called *tuples*.

The *schema of a relation* describes the structure of the relation. A *subschemata* describes only that portion of the database relevant for the user. For example: a librarian can see which books any patron has checked out, whereas the view of a patron is limited.

The *instance of a relation* is the set of tuples of the relation present at the database at any given moment.

# The Relational Model

some terminology

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A *relation* is a table with a fixed number of columns.

The columns are called *attributes*, taking values belonging to a *domain* (similar to types).

The rows are called *tuples*.

The *schema of a relation* describes the structure of the relation. A *subschemas* describes only that portion of the database relevant for the user. For example: a librarian can see which books any patron has checked out, whereas the view of a patron is limited.

The *instance of a relation* is the set of tuples of the relation present at the database at any given moment.

# The Relational Model

some terminology

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A *relation* is a table with a fixed number of columns.

The columns are called *attributes*, taking values belonging to a *domain* (similar to types).

The rows are called *tuples*.

The *schema of a relation* describes the structure of the relation. A *subschemas* describes only that portion of the database relevant for the user. For example: a librarian can see which books any patron has checked out, whereas the view of a patron is limited.

The *instance of a relation* is the set of tuples of the relation present at the database at any given moment.

# The Relational Model

some terminology

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A *relation* is a table with a fixed number of columns.

The columns are called *attributes*, taking values belonging to a *domain* (similar to types).

The rows are called *tuples*.

The *schema of a relation* describes the structure of the relation. A *subschemas* describes only that portion of the database relevant for the user. For example: a librarian can see which books any patron has checked out, whereas the view of a patron is limited.

The *instance of a relation* is the set of tuples of the relation present at the database at any given moment.

# The Relational Model

some terminology

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

A *relation* is a table with a fixed number of columns.

The columns are called *attributes*, taking values belonging to a *domain* (similar to types).

The rows are called *tuples*.

The *schema of a relation* describes the structure of the relation. A *subschemas* describes only that portion of the database relevant for the user. For example: a librarian can see which books any patron has checked out, whereas the view of a patron is limited.

The *instance of a relation* is the set of tuples of the relation present at the database at any given moment.

# Schema of our Library Database

## Databases

a library manager  
**relational design of  
databases**  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

In our library database, we have three relations:  
**Person, Book, and Check.**

Attributes of `Person` are identification number (`id`), name, (email) address, and status (librarian or patron).

Attributes of `Book` are identification number, author, title, and availability status (in or out).

`Check` relates identification numbers of people to the identification numbers of books checked out. The two attributes in `Check` are `person_id` and `book_id`.

# Schema of our Library Database

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

In our library database, we have three relations:  
**Person, Book, and Check.**

**Attributes of Person** are identification number (id), name, (email) address, and status (librarian or patron).

Attributes of **Book** are identification number, author, title, and availability status (in or out).

**Check** relates identification numbers of people to the identification numbers of books checked out. The two attributes in **Check** are `person_id` and `book_id`.

# Schema of our Library Database

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

In our library database, we have three relations:  
**Person, Book, and Check.**

**Attributes of Person** are identification number (id), name, (email) address, and status (librarian or patron).

**Attributes of Book** are identification number, author, title, and availability status (in or out).

**Check** relates identification numbers of people to the identification numbers of books checked out. The two attributes in **Check** are `person_id` and `book_id`.

# Schema of our Library Database

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

In our library database, we have three relations:  
**Person, Book, and Check.**

Attributes of **Person** are identification number (id), name, (email) address, and status (librarian or patron).

Attributes of **Book** are identification number, author, title, and availability status (in or out).

**Check** relates identification numbers of people to the identification numbers of books checked out. The two attributes in **Check** are `person_id` and `book_id`.

# Database Programming

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

## 1 Databases

a library manager  
relational design of databases  
**query, commit, rollback**

## 2 MySQL

an open source database  
running MySQL: database creation  
entering, modifying, and querying

## 3 MySQLdb: MySQL with Python

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# Structured Query Language (SQL)

Structured Query Language (SQL) is a standard supported on all relational databases.

Suppose we want to see the titles of all books checked out by a person with identification number equal to `nb`.

A query in SQL could then be formulated as

```
SELECT title
FROM BOOK, CHECK
WHERE CHECK.person_id = nb
      AND CHECK.book_id = BOOK.id
```

The result of this query is a new table, with one attribute: `title`.

# Structured Query Language (SQL)

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Structured Query Language (SQL) is a standard supported on all relational databases.

Suppose we want to see the titles of all books checked out by a person with identification number equal to `nb`.

A query in SQL could then be formulated as

```
SELECT title
FROM BOOK, CHECK
WHERE CHECK.person_id = nb
      AND CHECK.book_id = BOOK.id
```

The result of this query is a new table, with one attribute: `title`.

# Structured Query Language (SQL)

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Structured Query Language (SQL) is a standard supported on all relational databases.

Suppose we want to see the titles of all books checked out by a person with identification number equal to `nb`.

A query in SQL could then be formulated as

```
SELECT title
FROM BOOK, CHECK
WHERE CHECK.person_id = nb
      AND CHECK.book_id = BOOK.id
```

The result of this query is a new table, with one attribute: `title`.

# Maintaining Database Integrity

commit and rollback

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

For large, multiuser, and distributed databases (such as banking), losing data can be very costly.

To ensure database integrity, the information to perform the updated is backed up, stored in a log file.

The Commit/Rollback protocol regulates the update of a database in two stages:

**commit** At the commit point, all data to perform the update in the database has been stored in the log. In case of malfunction during the update, the data is retrieved from the log.

**rollback** If problem should arise during the actual update, the update can be made undone, using the information from the log.

# Maintaining Database Integrity

commit and rollback

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

For large, multiuser, and distributed databases (such as banking), losing data can be very costly.

To ensure database integrity, the information to perform the updated is backed up, stored in a log file.

The Commit/Rollback protocol regulates the update of a database in two stages:

**commit** At the commit point, all data to perform the update in the database has been stored in the log. In case of malfunction during the update, the data is retrieved from the log.

**rollback** If problem should arise during the actual update, the update can be made undone, using the information from the log.

# Maintaining Database Integrity

## commit and rollback

### Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

### MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

### MySQLdb: MySQL with Python

For large, multiuser, and distributed databases (such as banking), losing data can be very costly.

To ensure database integrity, the information to perform the updated is backed up, stored in a log file.

The Commit/Rollback protocol regulates the update of a database in two stages:

**commit** At the commit point, all data to perform the update in the database has been stored in the log. In case of malfunction during the update, the data is retrieved from the log.

**rollback** If problem should arise during the actual update, the update can be made undone, using the information from the log.

# Maintaining Database Integrity

## commit and rollback

### Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

### MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

### MySQLdb: MySQL with Python

For large, multiuser, and distributed databases (such as banking), losing data can be very costly.

To ensure database integrity, the information to perform the updated is backed up, stored in a log file.

The Commit/Rollback protocol regulates the update of a database in two stages:

**commit** At the commit point, all data to perform the update in the database has been stored in the log. In case of malfunction during the update, the data is retrieved from the log.

**rollback** If problem should arise during the actual update, the update can be made undone, using the information from the log.

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

# SQL is 4GL

structured query language

## Classification of programming languages:

- 1 native machine code for the CPU
- 2 assembler language
- 3 high level programming languages:  
Ada, C/C++, Fortran, Java, Python, etc...
- 4 languages closer to being like a human language  
→ SQL statements read like sentences.

SQL is fourth generation language (4GL).

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

# SQL is 4GL

structured query language

## Classification of programming languages:

- 1 native machine code for the CPU
- 2 assembler language
- 3 high level programming languages:  
Ada, C/C++, Fortran, Java, Python, etc...
- 4 languages closer to being like a human language  
→ SQL statements read like sentences.

SQL is fourth generation language (4GL).

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

# SQL is 4GL

structured query language

## Classification of programming languages:

- 1 native machine code for the CPU
- 2 assembler language
- 3 high level programming languages:  
Ada, C/C++, Fortran, Java, Python, etc...
- 4 languages closer to being like a human language  
→ SQL statements read like sentences.

SQL is fourth generation language (4GL).

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

# SQL is 4GL

structured query language

## Classification of programming languages:

- 1 native machine code for the CPU
- 2 assembler language
- 3 high level programming languages:  
Ada, C/C++, Fortran, Java, Python, etc...
- 4 languages closer to being like a human language  
→ SQL statements read like sentences.

SQL is fourth generation language (4GL).

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# SQL is 4GL

structured query language

## Classification of programming languages:

- 1 native machine code for the CPU
- 2 assembler language
- 3 high level programming languages:  
Ada, C/C++, Fortran, Java, Python, etc...
- 4 languages closer to being like a human language  
→ SQL statements read like sentences.

SQL is fourth generation language (4GL).

# Database Programming

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

**an open source  
database**  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

### 1 Databases

a library manager  
relational design of databases  
query, commit, rollback

### 2 MySQL

**an open source database**  
running MySQL: database creation  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

# MySQL & MySQLdb

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQL is an open source database,  
developed by the company MySQL AB.

In February 2008, Sun Microsystems acquired MySQL AB  
and the expertise of the GPL software for \$1 billion.

In January 2010, Oracle acquired Sun for \$7.38 billion.

MySQL can be downloaded for free from  
<http://www.mysql.com/downloads>.  
Following the instructions, install MySQL first.

MySQLdb is an interface to connect Python to MySQL.  
MySQLdb is an API (Application Programming Interface)  
like:

- 1 NumPy/SciPy for numerical/scientific computing
- 2 Tkinter for graphical user interfaces
- 3 the cgi module for server side web scripts

# MySQL & MySQLdb

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQL is an open source database,  
developed by the company MySQL AB.

In February 2008, Sun Microsystems acquired MySQL AB  
and the expertise of the GPL software for \$1 billion.

In January 2010, Oracle acquired Sun for \$7.38 billion.

MySQL can be downloaded for free from  
<http://www.mysql.com/downloads>.

Following the instructions, install MySQL first.

MySQLdb is an interface to connect Python to MySQL.  
MySQLdb is an API (Application Programming Interface)  
like:

- 1 NumPy/SciPy for numerical/scientific computing
- 2 Tkinter for graphical user interfaces
- 3 the cgi module for server side web scripts

# MySQL & MySQLdb

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQL is an open source database,  
developed by the company MySQL AB.

In February 2008, Sun Microsystems acquired MySQL AB  
and the expertise of the GPL software for \$1 billion.

In January 2010, Oracle acquired Sun for \$7.38 billion.

MySQL can be downloaded for free from  
<http://www.mysql.com/downloads>.

Following the instructions, install MySQL first.

MySQLdb is an interface to connect Python to MySQL.

MySQLdb is an API (Application Programming Interface)  
like:

- 1 NumPy/SciPy for numerical/scientific computing
- 2 Tkinter for graphical user interfaces
- 3 the cgi module for server side web scripts

# MySQL & MySQLdb

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQL is an open source database,  
developed by the company MySQL AB.

In February 2008, Sun Microsystems acquired MySQL AB  
and the expertise of the GPL software for \$1 billion.

In January 2010, Oracle acquired Sun for \$7.38 billion.

MySQL can be downloaded for free from  
<http://www.mysql.com/downloads>.

Following the instructions, install MySQL first.

MySQLdb is an interface to connect Python to MySQL.  
MySQLdb is an API (Application Programming Interface)  
like:

- 1 NumPy/SciPy for numerical/scientific computing
- 2 Tkinter for graphical user interfaces
- 3 the cgi module for server side web scripts

8 Mar 2010

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

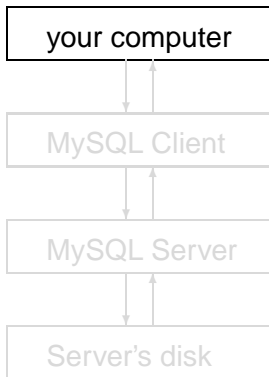
**an open source  
database**

running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# The Flow of Data

client/server computing



8 Mar 2010

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

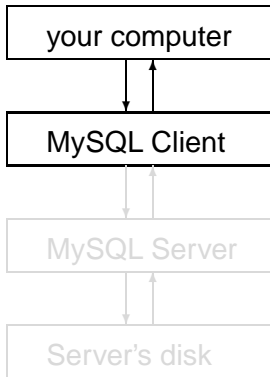
an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# The Flow of Data

client/server computing



8 Mar 2010

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

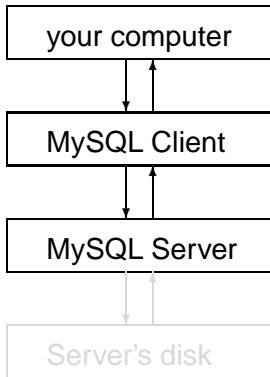
an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# The Flow of Data

client/server computing



8 Mar 2010

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

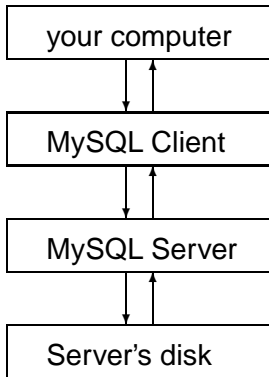
an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# The Flow of Data

client/server computing



# Database Programming

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
**running MySQL:  
database creation**  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

### 1 Databases

a library manager  
relational design of databases  
query, commit, rollback

### 2 MySQL

an open source database  
**running MySQL: database creation**  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

# Starting and Stopping the Daemon

as root on Mac OS X

It may be that MySQL is started at boot time.

Otherwise:

```
$ sudo mysqld_safe
```

```
Starting mysqld daemon with databases  
from /usr/local/mysql/data
```

Shutting the MySQL server down:

```
$ sudo mysqladmin shutdown
```

```
STOPPING server from pid file  
/usr/local/mysql/data/ambiorix.local.pid  
080304 21:33:10  mysqld ended
```

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

# Starting and Stopping the Daemon

as root on Mac OS X

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

It may be that MySQL is started at boot time.

Otherwise:

```
$ sudo mysqld_safe
```

```
Starting mysqld daemon with databases  
from /usr/local/mysql/data
```

Shutting the MySQL server down:

```
$ sudo mysqladmin shutdown
```

```
STOPPING server from pid file  
/usr/local/mysql/data/ambiorix.local.pid  
080304 21:33:10  mysqld ended
```

# Starting and Stopping the Daemon

as root on Mac OS X

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

It may be that MySQL is started at boot time.

Otherwise:

```
$ sudo mysqld_safe
```

```
Starting mysqld daemon with databases  
from /usr/local/mysql/data
```

Shutting the MySQL server down:

```
$ sudo mysqladmin shutdown
```

```
STOPPING server from pid file  
/usr/local/mysql/data/ambiorix.local.pid  
080304 21:33:10  mysqld ended
```

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# Running MySQL

## creating and deleting databases

The command `mysqladmin` is used in MySQL for server administration.

We need to use it to create first a database.

On a Mac OS X, at the prompt `$`:

```
$ sudo mysqladmin create Book
```

We have created a database with name `Book`.

To delete the database `Book`:

```
$ sudo mysqladmin drop Book
```

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# Running MySQL

## creating and deleting databases

The command `mysqladmin` is used in MySQL for server administration.

We need to use it to create first a database.

On a Mac OS X, at the prompt `$`:

```
$ sudo mysqladmin create Book
```

We have created a database with name `Book`.

To delete the database `Book`:

```
$ sudo mysqladmin drop Book
```

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# Running MySQL

## creating and deleting databases

The command `mysqladmin` is used in MySQL for server administration.

We need to use it to create first a database.

On a Mac OS X, at the prompt `$`:

```
$ sudo mysqladmin create Book
```

We have created a database with name `Book`.

To delete the database `Book`:

```
$ sudo mysqladmin drop Book
```

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

MySQLdb:  
MySQL with  
Python

# Running MySQL

## creating and deleting databases

The command `mysqladmin` is used in MySQL for server administration.

We need to use it to create first a database.

On a Mac OS X, at the prompt `$`:

```
$ sudo mysqladmin create Book
```

We have created a database with name `Book`.

To delete the database `Book`:

```
$ sudo mysqladmin drop Book
```

# MySQL to create the Table Book

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
$ sudo mysqladmin create Library
```

```
$ sudo mysql
```

```
Welcome to the MySQL monitor.  Commands end with ;  
Your MySQL connection id is 4  
Server version: 5.0.45 MySQL Community Server (GPL)
```

```
Type 'help;' or '\h' for help. Type '\c' to clear t
```

```
mysql> use Library;
```

```
Database changed
```

```
mysql> create table Book
```

```
    -> (id INT, title CHAR(80), available SMALLINT)
```

```
Query OK, 0 rows affected (0.00 sec)
```

We created a table Book with attributes

(1) id of domain INT; (2) title of domain CHAR,  
80 wide; and (3) available of domain SMALLINT.

8 Mar 2010

# MySQL to create the Table Book

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
$ sudo mysqladmin create Library
```

```
$ sudo mysql
```

```
Welcome to the MySQL monitor.  Commands end with ;
```

```
Your MySQL connection id is 4
```

```
Server version: 5.0.45 MySQL Community Server (GPL)
```

```
Type 'help;' or '\h' for help. Type '\c' to clear t
```

```
mysql> use Library;
```

```
Database changed
```

```
mysql> create table Book
```

```
    -> (id INT, title CHAR(80), available SMALLINT)
```

```
Query OK, 0 rows affected (0.00 sec)
```

We created a table `Book` with attributes

(1) `id` of domain `INT`; (2) `title` of domain `CHAR`,  
80 wide; and (3) `available` of domain `SMALLINT`.

8 Mar 2010

# MySQL to create the Table Book

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
$ sudo mysqladmin create Library
```

```
$ sudo mysql
```

```
Welcome to the MySQL monitor.  Commands end with ;
```

```
Your MySQL connection id is 4
```

```
Server version: 5.0.45 MySQL Community Server (GPL)
```

```
Type 'help;' or '\h' for help. Type '\c' to clear t
```

```
mysql> use Library;
```

```
Database changed
```

```
mysql> create table Book
```

```
    -> (id INT, title CHAR(80), available SMALLINT)
```

```
Query OK, 0 rows affected (0.00 sec)
```

**We created a table Book with attributes**

(1) id of domain INT; (2) title of domain CHAR,  
80 wide; and (3) available of domain SMALLINT.

# MySQL to create the Table Book

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
$ sudo mysqladmin create Library
```

```
$ sudo mysql
```

```
Welcome to the MySQL monitor.  Commands end with ;
```

```
Your MySQL connection id is 4
```

```
Server version: 5.0.45 MySQL Community Server (GPL)
```

```
Type 'help;' or '\h' for help. Type '\c' to clear t
```

```
mysql> use Library;
```

```
Database changed
```

```
mysql> create table Book
```

```
    -> (id INT, title CHAR(80), available SMALLINT)
```

```
Query OK, 0 rows affected (0.00 sec)
```

We created a table `Book` with attributes

(1) `id` of domain `INT`; (2) `title` of domain `CHAR`,  
80 wide; and (3) `available` of domain `SMALLINT`.

# MySQL to create the Table Book

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
$ sudo mysqladmin create Library
```

```
$ sudo mysql
```

```
Welcome to the MySQL monitor.  Commands end with ;
```

```
Your MySQL connection id is 4
```

```
Server version: 5.0.45 MySQL Community Server (GPL)
```

```
Type 'help;' or '\h' for help. Type '\c' to clear t
```

```
mysql> use Library;
```

```
Database changed
```

```
mysql> create table Book
```

```
    -> (id INT, title CHAR(80), available SMALLINT)
```

```
Query OK, 0 rows affected (0.00 sec)
```

We created a table `Book` with attributes

(1) `id` of domain `INT`; (2) `title` of domain `CHAR`,  
80 wide; and (3) `available` of domain `SMALLINT`.

# The Tables Person and Checked

## MySQL for the database Library

### Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

### MySQL

an open source  
database

running MySQL:  
database creation

entering, modifying,  
and querying

### MySQLdb: MySQL with Python

```
mysql> create table Person
      -> (id INT, name CHAR(20), status SMALLINT);
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> create table Checked
      -> (idbook INT, idname INT);
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> show tables;
+-----+
| Tables_in_library |
+-----+
| Book               |
| Checked            |
| Person             |
+-----+
3 rows in set (0.01 sec)
```

8 Mar 2010

# The Tables Person and Checked

MySQL for the database Library

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> create table Person
      -> (id INT, name CHAR(20), status SMALLINT);
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> create table Checked
      -> (idbook INT, idname INT);
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> show tables;
+-----+
| Tables_in_library |
+-----+
| Book               |
| Checked            |
| Person             |
+-----+
3 rows in set (0.01 sec)
```

# Database Programming

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

### 1 Databases

a library manager  
relational design of databases  
query, commit, rollback

### 2 MySQL

an open source database  
running MySQL: database creation  
entering, modifying, and querying

### 3 MySQLdb: MySQL with Python

8 Mar 2010

# Entering Data

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Book;
```

Field	Type	Null	Key	Default
id	int(11)	YES		NULL
title	char(80)	YES		NULL
available	smallint(6)	YES		NULL

```
mysql> insert into Book values
```

```
    -> (1,"The Art & Craft of Computing",1);
```

```
mysql> insert into Book values
```

```
    -> (2,"Making Use of Python",1);
```

```
mysql> select * from Book;
```

id	title	available
1	The Art & Craft of Computing	1
2	Making Use of Python	1

8 Mar 2010

# Entering Data

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Book;
```

Field	Type	Null	Key	Default
id	int(11)	YES		NULL
title	char(80)	YES		NULL
available	smallint(6)	YES		NULL

```
mysql> insert into Book values
```

```
    -> (1,"The Art & Craft of Computing",1);
```

```
mysql> insert into Book values
```

```
    -> (2,"Making Use of Python",1);
```

```
mysql> select * from Book;
```

id	title	available
1	The Art & Craft of Computing	1
2	Making Use of Python	1

8 Mar 2010

# Entering Data

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Book;
```

Field	Type	Null	Key	Default
id	int(11)	YES		NULL
title	char(80)	YES		NULL
available	smallint(6)	YES		NULL

```
mysql> insert into Book values
```

```
    -> (1,"The Art & Craft of Computing",1);
```

```
mysql> insert into Book values
```

```
    -> (2,"Making Use of Python",1);
```

```
mysql> select * from Book;
```

id	title	available
1	The Art & Craft of Computing	1
2	Making Use of Python	1

8 Mar 2010

# Inserting Records in Person

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Ext
id	int(11)	YES		NULL	
name	char(20)	YES		NULL	
status	smallint(6)	YES		NULL	

```
mysql> insert into Person values
```

```
    -> (1, "Rashi Gupta", 1);
```

```
mysql> insert into Person values
```

```
    -> (2, "Guido van Rossum", 0);
```

```
mysql> select * from Person;
```

id	name	status
1	Rashi Gupta	1
2	Guido van Rossum	0

8 Mar 2010

# Inserting Records in Person

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Ext
id	int(11)	YES		NULL	
name	char(20)	YES		NULL	
status	smallint(6)	YES		NULL	

```
mysql> insert into Person values
```

```
    -> (1, "Rashi Gupta", 1);
```

```
mysql> insert into Person values
```

```
    -> (2, "Guido van Rossum", 0);
```

```
mysql> select * from Person;
```

id	name	status
1	Rashi Gupta	1
2	Guido van Rossum	0

8 Mar 2010

# Inserting Records in Person

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Ext
id	int(11)	YES		NULL	
name	char(20)	YES		NULL	
status	smallint(6)	YES		NULL	

```
mysql> insert into Person values
```

```
    -> (1, "Rashi Gupta", 1);
```

```
mysql> insert into Person values
```

```
    -> (2, "Guido van Rossum", 0);
```

```
mysql> select * from Person;
```

id	name	status
1	Rashi Gupta	1
2	Guido van Rossum	0

# Inserting Records in Checked

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

```
mysql> explain Checked;
```

Field	Type	Null	Key	Default	Extra
idbook	int(11)	YES		NULL	
idname	int(11)	YES		NULL	

```
mysql> insert into Checked values (1,2);
```

```
mysql> select * from Checked;
```

idbook	idname
1	2

# Inserting Records in Checked

## Databases

a library manager  
 relational design of  
 databases  
 query, commit,  
 rollback

## MySQL

an open source  
 database  
 running MySQL:  
 database creation  
 entering, modifying,  
 and querying

## MySQLdb: MySQL with Python

```
mysql> explain Checked;
```

Field	Type	Null	Key	Default	Extra
idbook	int(11)	YES		NULL	
idname	int(11)	YES		NULL	

```
mysql> insert into Checked values (1,2);
```

```
mysql> select * from Checked;
```

idbook	idname
1	2

# Inserting Records in Checked

## Databases

a library manager  
 relational design of  
 databases  
 query, commit,  
 rollback

## MySQL

an open source  
 database  
 running MySQL:  
 database creation  
 entering, modifying,  
 and querying

## MySQLdb: MySQL with Python

```
mysql> explain Checked;
```

Field	Type	Null	Key	Default	Extra
idbook	int(11)	YES		NULL	
idname	int(11)	YES		NULL	

```
mysql> insert into Checked values (1,2);
```

```
mysql> select * from Checked;
```

idbook	idname
1	2

# Modifying Records in Book

## Databases

a library manager  
 relational design of  
 databases  
 query, commit,  
 rollback

## MySQL

an open source  
 database  
 running MySQL:  
 database creation  
 entering, modifying,  
 and querying

## MySQLdb: MySQL with Python

The book with id = 1 is no longer available:

```
mysql> update Book set available=0 where id = 1;
Query OK, 1 row affected (0.00 sec)
```

```
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> select * from Book;
```

id	title	available
1	The Art & Craft of Computing	0
2	Making Use of Python	1

# Modifying Records in Book

## Databases

a library manager  
 relational design of  
 databases  
 query, commit,  
 rollback

## MySQL

an open source  
 database  
 running MySQL:  
 database creation  
 entering, modifying,  
 and querying

## MySQLdb: MySQL with Python

The book with id = 1 is no longer available:

```
mysql> update Book set available=0 where id = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> select * from Book;
```

id	title	available
1	The Art & Craft of Computing	0
2	Making Use of Python	1

# A more involved Query

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Let us select the titles of all books  
Guido van Rossum has checked out:

```
mysql> select title
-> from Book, Person, Checked
-> where Person.name = "Guido van Rossum"
->    and Checked.idname = Person.id
->    and Checked.idbook = Book.id;
```

```
+-----+
| title |
+-----+
| The Art & Craft of Computing |
+-----+
```

```
mysql> exit;
Bye
```

# A more involved Query

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Let us select the titles of all books  
Guido van Rossum has checked out:

```
mysql> select title
-> from Book, Person, Checked
-> where Person.name = "Guido van Rossum"
->    and Checked.idname = Person.id
->    and Checked.idbook = Book.id;
```

```
+-----+
| title |
+-----+
| The Art & Craft of Computing |
+-----+
```

```
mysql> exit;
Bye
```

# A more involved Query

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Let us select the titles of all books  
Guido van Rossum has checked out:

```
mysql> select title
-> from Book, Person, Checked
-> where Person.name = "Guido van Rossum"
->    and Checked.idname = Person.id
->    and Checked.idbook = Book.id;
```

```
+-----+
| title |
+-----+
| The Art & Craft of Computing |
+-----+
```

```
mysql> exit;
Bye
```

# A more involved Query

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

Let us select the titles of all books  
Guido van Rossum has checked out:

```
mysql> select title
-> from Book, Person, Checked
-> where Person.name = "Guido van Rossum"
->    and Checked.idname = Person.id
->    and Checked.idbook = Book.id;
```

```
+-----+
| title |
+-----+
| The Art & Craft of Computing |
+-----+
```

```
mysql> exit;
Bye
```

# Using MySQLdb in Python

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQLdb is an interface to use MySQL  
from within a Python session.

At the command prompt \$:

```
$ sudo python
Python 2.5.1 (r251:54869, Apr 18 2007, 22:08:04)
[GCC 4.0.1 (Apple Computer, Inc. build 5367)] on da
Type "help", "copyright", "credits" or "license" fo
>>> import MySQLdb
>>> L = MySQLdb.connect(db="Library")
>>> c = L.cursor()
```

Observe:

- run Python as superuser, otherwise no access;
- with connect(), we identify the database Library;
- L.cursor() returns a new object to represent a database cursor used to manage all operations.

# Using MySQLdb in Python

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQLdb is an interface to use MySQL  
from within a Python session.

At the command prompt \$:

```
$ sudo python
Python 2.5.1 (r251:54869, Apr 18 2007, 22:08:04)
[GCC 4.0.1 (Apple Computer, Inc. build 5367)] on da
Type "help", "copyright", "credits" or "license" fo
>>> import MySQLdb
>>> L = MySQLdb.connect(db="Library")
>>> c = L.cursor()
```

Observe:

- run Python as superuser, otherwise no access;
- with connect(), we identify the database Library;
- L.cursor() returns a new object to represent a database cursor used to manage all operations.

# Using MySQLdb in Python

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQLdb is an interface to use MySQL  
from within a Python session.

At the command prompt \$:

```
$ sudo python
```

```
Python 2.5.1 (r251:54869, Apr 18 2007, 22:08:04)  
[GCC 4.0.1 (Apple Computer, Inc. build 5367)] on da  
Type "help", "copyright", "credits" or "license" fo
```

```
>>> import MySQLdb
```

```
>>> L = MySQLdb.connect(db="Library")
```

```
>>> c = L.cursor()
```

Observe:

- run Python as superuser, otherwise no access;
- with `connect()`, we identify the database `Library`;
- `L.cursor()` returns a new object to represent a database cursor used to manage all operations.

# Using MySQLdb in Python

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQLdb is an interface to use MySQL  
from within a Python session.

At the command prompt \$:

```
$ sudo python
```

```
Python 2.5.1 (r251:54869, Apr 18 2007, 22:08:04)  
[GCC 4.0.1 (Apple Computer, Inc. build 5367)] on da  
Type "help", "copyright", "credits" or "license" fo
```

```
>>> import MySQLdb
```

```
>>> L = MySQLdb.connect(db="Library")
```

```
>>> c = L.cursor()
```

Observe:

- run Python as superuser, otherwise no access;
- with `connect()`, we identify the database `Library`;
- `L.cursor()` returns a new object to represent a database cursor used to manage all operations.

# Using MySQLdb in Python

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

MySQLdb is an interface to use MySQL  
from within a Python session.

At the command prompt \$:

```
$ sudo python
```

```
Python 2.5.1 (r251:54869, Apr 18 2007, 22:08:04)  
[GCC 4.0.1 (Apple Computer, Inc. build 5367)] on da  
Type "help", "copyright", "credits" or "license" fo
```

```
>>> import MySQLdb
```

```
>>> L = MySQLdb.connect(db="Library")
```

```
>>> c = L.cursor()
```

Observe:

- run Python as superuser, otherwise no access;
- with `connect()`, we identify the database `Library`;
- `L.cursor()` returns a new object to represent a database cursor used to manage all operations.

# Retrieving Information

using `execute` and `fetchone`

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With a cursor `c`, we pass MySQL commands as string argument to the `c.execute()` method:

```
>>> c.execute("show tables")
3L
```

The `3L` indicates there are 3 lines of output. To retrieve the output line by line, we use `fetchone`:

```
>>> c.fetchone()
('Book',)
>>> c.fetchone()
('Checked',)
>>> c.fetchone()
('Person',)
>>> c.fetchone()
```

# Retrieving Information

using `execute` and `fetchone`

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With a cursor `c`, we pass MySQL commands as string argument to the `c.execute()` method:

```
>>> c.execute("show tables")  
3L
```

The `3L` indicates there are 3 lines of output. To retrieve the output line by line, we use `fetchone`:

```
>>> c.fetchone()  
( 'Book' , )  
>>> c.fetchone()  
( 'Checked' , )  
>>> c.fetchone()  
( 'Person' , )  
>>> c.fetchone()
```

# Retrieving Information

using `execute` and `fetchone`

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With a cursor `c`, we pass MySQL commands as string argument to the `c.execute()` method:

```
>>> c.execute("show tables")
3L
```

The `3L` indicates there are 3 lines of output.

To retrieve the output line by line, we use `fetchone`:

```
>>> c.fetchone()
('Book',)
>>> c.fetchone()
('Checked',)
>>> c.fetchone()
('Person',)
>>> c.fetchone()
```

# Retrieving Information

using `execute` and `fetchone`

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With a cursor `c`, we pass MySQL commands as string argument to the `c.execute()` method:

```
>>> c.execute("show tables")  
3L
```

The `3L` indicates there are 3 lines of output. To retrieve the output line by line, we use `fetchone`:

```
>>> c.fetchone()  
( 'Book' , )  
>>> c.fetchone()  
( 'Checked' , )  
>>> c.fetchone()  
( 'Person' , )  
>>> c.fetchone()
```

# Retrieving Information

using `execute` and `fetchone`

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With a cursor `c`, we pass MySQL commands as string argument to the `c.execute()` method:

```
>>> c.execute("show tables")
3L
```

The `3L` indicates there are 3 lines of output. To retrieve the output line by line, we use `fetchone`:

```
>>> c.fetchone()
('Book',)
>>> c.fetchone()
('Checked',)
>>> c.fetchone()
('Person',)
>>> c.fetchone()
```

# All Results at Once

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With `fetchall()` we obtain all results at once.

```
>>> c.execute("select * from Book")
2L
>>> c.fetchall()
((1L, 'The Art & Craft of Computing', 0),\
 (2L, 'Making Use of Python', 1))
```

A more involved query:

```
>>> s = "select title " + \
... " from Book, Person, Checked" + \
... " where Person.Name = \"Guido van Rossum\" " + \
... " and Checked.idname = Person.id" + \
... " and Checked.idbook = Book.id"
>>> r = c.execute(s)
>>> c.fetchall()
(('The Art & Craft of Computing',),)
```

# All Results at Once

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With `fetchall()` we obtain all results at once.

```
>>> c.execute("select * from Book")
```

```
2L
```

```
>>> c.fetchall()
```

```
((1L, 'The Art & Craft of Computing', 0),\
 (2L, 'Making Use of Python', 1))
```

A more involved query:

```
>>> s = "select title " + \
... " from Book, Person, Checked" + \
... " where Person.Name = \"Guido van Rossum\" " + \
... " and Checked.idname = Person.id" + \
... " and Checked.idbook = Book.id"
>>> r = c.execute(s)
>>> c.fetchall()
(('The Art & Craft of Computing',),)
```

# All Results at Once

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With `fetchall()` we obtain all results at once.

```
>>> c.execute("select * from Book")
2L
>>> c.fetchall()
((1L, 'The Art & Craft of Computing', 0),\
 (2L, 'Making Use of Python', 1))
```

A more involved query:

```
>>> s = "select title " + \
... " from Book, Person, Checked" + \
... " where Person.Name = \"Guido van Rossum\" " + \
... " and Checked.idname = Person.id" + \
... " and Checked.idbook = Book.id"
>>> r = c.execute(s)
>>> c.fetchall()
(('The Art & Craft of Computing',),)
```

# All Results at Once

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With `fetchall()` we obtain all results at once.

```
>>> c.execute("select * from Book")
2L
>>> c.fetchall()
((1L, 'The Art & Craft of Computing', 0),\
 (2L, 'Making Use of Python', 1))
```

A more involved query:

```
>>> s = "select title " + \
... " from Book, Person, Checked" + \
... " where Person.Name = \"Guido van Rossum\" " + \
... " and Checked.idname = Person.id" + \
... " and Checked.idbook = Book.id"
>>> r = c.execute(s)
>>> c.fetchall()
(('The Art & Craft of Computing',),)
```

# All Results at Once

## Databases

a library manager  
relational design of  
databases

query, commit,  
rollback

## MySQL

an open source  
database

running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

With `fetchall()` we obtain all results at once.

```
>>> c.execute("select * from Book")
2L
>>> c.fetchall()
((1L, 'The Art & Craft of Computing', 0),\
 (2L, 'Making Use of Python', 1))
```

A more involved query:

```
>>> s = "select title " + \
... " from Book, Person, Checked" + \
... " where Person.Name = \"Guido van Rossum\" " + \
... " and Checked.idname = Person.id" + \
... " and Checked.idbook = Book.id"
>>> r = c.execute(s)
>>> c.fetchall()
(('The Art & Craft of Computing',),)
```

# Summary + Assignments

## Databases

a library manager  
relational design of  
databases  
query, commit,  
rollback

## MySQL

an open source  
database  
running MySQL:  
database creation  
entering, modifying,  
and querying

## MySQLdb: MySQL with Python

We started chapter 11 in *Making Use of Python*;  
see also §9.1,2 in *Computer Science, an overview*,  
visit <http://www.mysqltutorial.org/>.

## Assignments:

- 1 Design a simple relational database to manage bank accounts, and to perform financial transactions.
- 2 Download and install MySQL and MySQLdb.
- 3 Use MySQL to create a database to perform simple financial transactions, implementing the design of assignment 1.
- 4 Write a Python function that takes on input the name of a person and returns the query to select all titles of the books that person has checked out.