

Databases

- the library manager revisited
- relational design of databases
- query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

- the library manager revisited
- relational design of databases
- query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

MCS 260 Lecture 37
Introduction to Computer Science
Jan Vershelde, 21 November 2007

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

MySQL

an open source database
running MySQL: database
creation

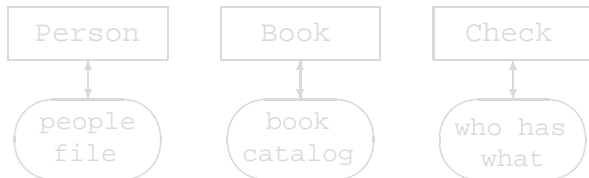
entering, modifying, and
querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

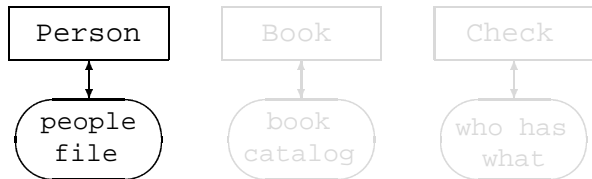
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

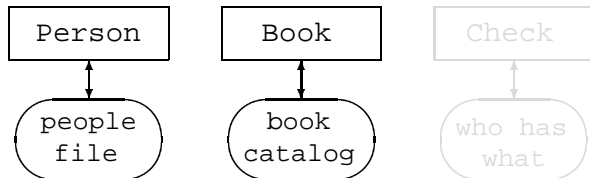
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

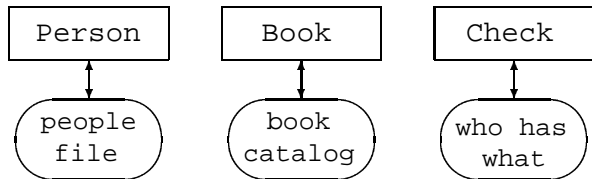
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

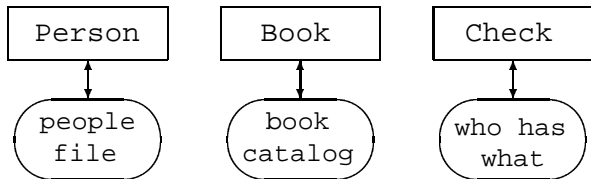
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

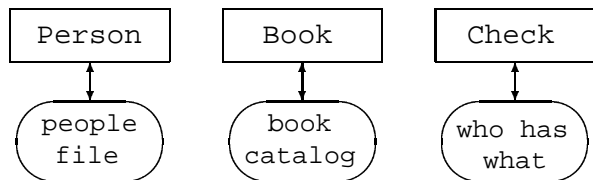
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

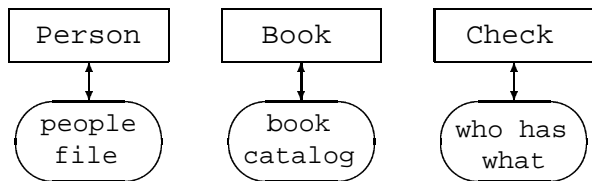
entering, modifying, and querying

MySQLdb: MySQL with Python

File Oriented Information Systems

the library manager revisited

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

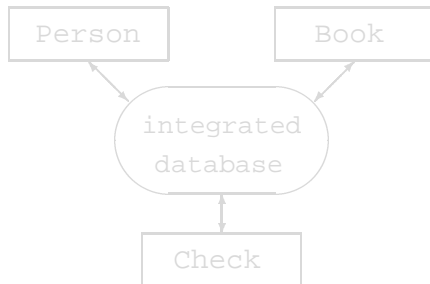
an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Database Information Systems

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*.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

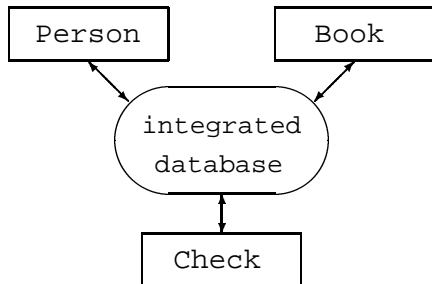
an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Database Information Systems

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*.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited

**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 Relational Model

some terminology

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.

Databases

the library manager
revisited

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 Relational Model

some terminology

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.

Databases

the library manager revisited

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 Relational Model

some terminology

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.

Databases

the library manager revisited

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 Relational Model

some terminology

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.

Databases

the library manager revisited

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 Relational Model

some terminology

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.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Schema of our Library Database

MCS 260 L-37

21 November 2007

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`.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

MySQL

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQLdb: MySQL
with Python

Schema of our Library Database

MCS 260 L-37

21 November 2007

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`.

Databases

the library manager
revisited

**relational design of
databases**

query, commit, rollback

MySQL

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQLdb: MySQL
with Python

Schema of our Library Database

MCS 260 L-37

21 November 2007

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`.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

MySQL

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQLdb: MySQL
with Python

Schema of our Library Database

MCS 260 L-37

21 November 2007

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`.

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
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`.

Databases

the library manager revisited
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`.

Databases

the library manager revisited
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`.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Maintaining Database Integrity

commit and rollback

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

To ensure database integrity, the information to perform the update 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

the library manager
revisited

relational design of
databases

query, commit, rollback

MySQL

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQLdb: MySQL with Python

Maintaining Database Integrity

commit and rollback

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

To ensure database integrity, the information to perform the update 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

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Maintaining Database Integrity

commit and rollback

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

To ensure database integrity, the information to perform the update 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

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Maintaining Database Integrity

commit and rollback

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

To ensure database integrity, the information to perform the update 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

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Data Mining

databases + statistics

MCS 260 L-37

21 November 2007

A relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Data Mining

databases + statistics

MCS 260 L-37

21 November 2007

A relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Data Mining

databases + statistics

MCS 260 L-37

21 November 2007

A relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager revisited
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 relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager revisited
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 relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager revisited
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 relatively new field is *data mining*, combining database technology and statistics.

In data mining one develops techniques to discover patterns in large collections of data. Another related discipline is *machine learning*.

Some examples of application areas:

- ▶ investment analysis, e.g: predict stock performance
- ▶ market trends, such as correlate purchase behavior
- ▶ machine learning to play chess
- ▶ decode DNA, identify genes that cause a disease

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
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

an open source database

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

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.

Databases

the library manager
revisited
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

an open source database

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

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.

Databases

the library manager
revisited
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

an open source database

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

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.

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
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

the library manager
revisited
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

the library manager
revisited
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

the library manager
revisited
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

the library manager
revisited
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 to create the Table Book

MCS 260 L-37

21 November 2007

```
$ 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
```

```
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.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

MySQL \g:

an open source database
running MySQL: database
creation

entering, modifying, and
querying

The MySQL
MySQLdb: MySQL
with Python

MySQL to create the Table Book

MCS 260 L-37

21 November 2007

```
$ sudo mysqladmin create Library
$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
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 the buffer.

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`.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQL Sub: MySQL
with Python

MySQL to create the Table Book

MCS 260 L-37

21 November 2007

```
$ sudo mysqladmin create Library
$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
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 the buffer.

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.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQL: MySQL with Python

MySQL to create the Table Book

MCS 260 L-37

21 November 2007

```
$ sudo mysqladmin create Library
$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
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 the buffer.

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.

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

an open source database

running MySQL: database
creation

entering, modifying, and
querying

MySQL Sub: MySQL
with Python

MySQL to create the Table Book

MCS 260 L-37

21 November 2007

Databases

the library manager
revisited

relational design of
databases

query, commit, rollback

an open source database
running MySQL: database
creation

entering, modifying, and
querying

MySQL Sub: MySQL
with Python

```
$ sudo mysqladmin create Library
$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
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 the buffer.

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

```
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)
```

Databases

the library manager revisited
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 Tables Person and Checked

MySQL for the database Library

```
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)
```

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

21 November 2007

Entering Data

```
mysql> explain Book;
```

Field	Type	Null	Key	Default	Extra
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

Databases

the library manager revisited

relational databases

query, commit, rollback

MySQL

an open source database

running MySQL: database

creation

entering, modifying, and

querying

MySQLdb: MySQL

with Python

21 November 2007

Entering Data

```
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

Databases

the library manager revisited

relational databases

query, commit, rollback

MySQL

an open source database

running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

21 November 2007

Entering Data

```
mysql> explain Book;
```

Field	Type	Null	Key	Default	Extra
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

Databases

the library manager revisited

relational databases

query commit rollback

MySQL

an open source database

running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Inserting Records in Person

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Extra
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

Databases

the library manager revisited

relational design of databases

query commit rollback

MySQL

an open source database running MySQL database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Inserting Records in Person

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Extra
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

Databases

the library manager revisited

the logical design of databases

query, commit, rollback

MySQL

an open source database running MySQL: database creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Inserting Records in Person

```
mysql> explain Person;
```

Field	Type	Null	Key	Default	Extra
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

Databases

the library manager revisited

relational design of databases

query, commit, rollback

MySQL

an open source database

running MySQL: database

creation

entering, modifying, and querying

MySQLdb: MySQL with Python

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Inserting Records in Checked

```
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

```
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

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Inserting Records in Checked

```
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

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

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Modifying Records in Book

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

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
```

Databases

the library manager
revisited
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 more involved Query

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
```

Databases

the library manager
revisited
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 more involved Query

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
```

Databases

the library manager
revisited
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 more involved Query

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
```

Databases

the library manager
 revisited
 relational design of
 databases
 query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Using MySQLdb in 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
Type "help", "copyright", "credits" or "license"
>>> 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.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Using MySQLdb in 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 darwin
Type "help", "copyright", "credits" or "license"
>>> 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.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

an open source database running MySQL: database creation, creating, inserting, and querying

for more:
[MySQLdb: MySQL with Python](#)

Using MySQLdb in 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 darwin
Type "help", "copyright", "credits" or "license"
>>> 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.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

an open source database running MySQL: database creation, creating, inserting, and querying

darwin
for more :
[MySQLdb: MySQL with Python](#)

Using MySQLdb in 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 darwin
Type "help", "copyright", "credits" or "license"
>>> 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.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

an open source database running MySQL: database creation, creating, inserting, and querying

for more :
[MySQLdb: MySQL with Python](#)

Using MySQLdb in 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 darwin
Type "help", "copyright", "credits" or "license"
>>> 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.

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

an open source database running MySQL: database creation, creating, inserting, and querying

for more :
[MySQLdb: MySQL with Python](#)

Retrieving Information

using `execute` and `fetchone`

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()
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Retrieving Information

using `execute` and `fetchone`

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()
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Retrieving Information

using `execute` and `fetchone`

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()
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Retrieving Information

using `execute` and `fetchone`

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()
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL
with Python

Retrieving Information

using `execute` and `fetchone`

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()
```

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

All Results at Once

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',),)
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

All Results at Once

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',),)
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

All Results at Once

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',),)
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

All Results at Once

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',),)
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

All Results at Once

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',),)
```

Databases

the library manager
revisited
relational design of
databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python

Summary + Assignments

We started chapter 11 in *Making Use of Python*; see also §16.3 in *The Art & Craft of Computing*.

Assignments:

1. Design a simple relational database to manage bank accounts, and to perform financial transactions.
2. Download and install MySQL.
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.

Happy Thanksgiving!

Databases

the library manager revisited
relational design of databases
query, commit, rollback

MySQL

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

MySQLdb: MySQL with Python