The events flow through the replication system from the master to the slaves in the

following manner:

1.The session accepts a statement from the client, executes the statement, and syn-

chronizes with other sessions to ensure each transaction is executed without con-

flicting with other changes made by other sessions.

2.Just before the statement finishes execution, an entry consisting of one or more

events is written to the binary log. This process is covered in Chapter 2 and will

not be described again in this chapter.

3.After the events have been written to the binary log, a dump thread in the master

takes over, reads the events from the binary log, and sends them over to the slave’s

I/O thread.

4.When the slave I/O thread receives the event, it writes it to the end of the relay log.

5.Once in the relay log, a slave SQL thread reads the event from the relay log and

executes the event to apply the changes to the database on the slave.

If the connection to the master is lost, the slave I/O thread will try to reconnect to the

server in the same way that any MySQL client thread does. Some of the options that

The Structure of the Relay Log

As the previous section shows, the relay log is the information that ties the master and

slave together—the heart of replication. It’s important to be aware of how it is used

and how the slave threads coordinate through it. Therefore, we’ll go through the details

here of how the relay log is structured and how the slave threads use the relay log to

handle replication.

As described in the previous section, the events sent from the master are stored in the

relay log by the I/O thread. The relay log serves as a buffer so that the master does not

have to wait for the slave execution to finish before sending the next event.

Figure 6-2 shows a schematic view of the relay log. It’s similar in structure to the binlog

on the master but has some extra files.Figure 6-1. Master and several slaves with internal architecture

Replication

In addition to the content files and the index files in the binary log, the relay log also

maintains two files to keep track of replication progress: the relay log information file

and the master log information file. The names of these two files are controlled by two

options in the my.cnf file:

relay-log-info-file=filename

This option sets the name of the relay log information file. It is also available as the

read-only server variable relay\_log\_info\_file. Unless an absolute filename is

given, the filename is relative to the data directory of the server. The default file-

name is relay-log.info.

master-info-file=filename

This option sets the name of the master log information file. The default filename

is master.info.

The information in the master.info file takes precedence over informa-

tion in the my.cnf file. This means that if you change information in the

my.cnf file and restart the server, the information will still be read from

the master.info file instead of from the my.cnf file.

For this reason, the recommendation is not to put any of the options

that can be specified with the CHANGE MASTER TO command in the

my.cnf file, but instead to use the CHANGE MASTER TO command to con-

figure replication. If, for some reason, you want to put any of the repli-

cation options in the my.cnf file and you want to make sure that the

options are read from it when starting the slave, you have to issue RESET

SLAVE before editing the my.cnf file.

Beware when executing RESET SLAVE! It will delete the master.info file,

the relay-log.info file, and all the relay logfiles!

The password is written unencrypted in the master.info file. For that

reason, it is critical to protect the file so it can be read only by the MySQL

server. The standard way to ensure this is to define a dedicated user on

the server to run the server, assign all the files responsible for replication

and database maintenance to this user, and remove all permissions from

the files except read and write by this user.