```
import java.rmi.*;
import java.rmi.server.*;

public interface RemoteDatabase extends Remote {
 public int read() throws RemoteException;
 public void write(int value) throws RemoteException;
}

class Client {
 public static void main(String[] args) {
  try {
   // set the standard RMI security manager
   System.setSecurityManager(new RMISecurityManager());

   // get remote database object
   String name =
       "rmi://paloverde:9999/database";
   RemoteDatabase db =
       (RemoteDatabase) Naming.lookup(name);

   // read command-line argument and access database
   int value, rounds = Integer.parseInt(args[0]);
   for (int i = 0; i < rounds; i++) {
     value = db.read();
     System.out.println("read: " + value);
     db.write(value+1);
   }
  }
  catch (Exception e) {
    System.err.println(e);
  }
 }
}

class RemoteDatabaseServer extends UnicastRemoteObject
          implements RemoteDatabase {
 protected int data = 0;  // the "database"

 public int read() throws RemoteException {
   return data;
 }

 public void write(int value) throws RemoteException {
   data = value;
   System.out.println("new value is: " + data);
 }

 // constructor required because of throws clause
 public RemoteDatabaseServer() throws RemoteException {
   super();
 }
```

```
 public static void main(String[] args) {
  try {
   // create a remote database server object
   RemoteDatabaseServer server =
       new RemoteDatabaseServer();
   // register name and start serving!
   String name =
       "rmi://paloverde:9999/database";
   Naming.bind(name, server);
   System.out.println(name + " is running");
  }
  catch (Exception e) {
    System.err.println(e);
  }
 }
}
```

Figure 8.16 Remote database interface, client, and server.