39
Database Administration

This chapter discusses the database administration methods introduced in Oracle Database 11*g* Release 1. This chapter contains the following sections:

39.1 Using the Database Administration Methods

Starting from Oracle Database 11*g* Release 1, two JDBC methods, `startup` and `shutdown`, has been added in the `oracle.jdbc.OracleConnection` interface, which enable you to start up and shut down an Oracle Database instance. This is similar to the way you would start up or shut down a database instance from SQL*Plus.

To use the `startup` and `shutdown` methods, you must:

- Have a dedicated connection to the server. You cannot be connected to a shared server through a dispatcher.
- Be connected as `SYSDBA` or `SYSOPER`. To connect as `SYSDBA` or `SYSOPER` with Oracle JDBC drivers, you need to set the `INTERNAL_LOGON` connection property accordingly.

To log on as `SYSDBA` with the JDBC Thin driver you must configure the server to use the password file. For example, to configure `system/manager` to connect as `SYSDBA` with the JDBC Thin driver, perform the following:

1. From the command line, type:

```
orapwd file=$ORACLE_HOME/dbs/orapw entries=5
Enter password: password
```

2. Connect to database as `SYSDBA` and run the following commands from SQL*Plus:

```
GRANT SYSDBA TO system;
PASSWORD system
       Changing password for system
       New password: password
       Retype new password: password
```

3. Edit `init.ora` and add the following line:

```
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
```

4. Restart the database instance.

As opposed to the JDBC Thin driver, the JDBC OCI driver can connect as `SYSDBA` or `SYSOPER` locally without specifying a password file on the server.

39.2 Using the startup Method

To start a database instance using the `startup` method, the application must first connect to the database as a `SYSDBA` or `SYSOPER` in the `PRELIM_AUTH` mode, which is the only connection mode that is permitted when the database is down. You can do this by setting the connection property `PRELIM_AUTH` to `true`. In the `PRELIM_AUTH` mode, you can *only* start up a database instance that is down. You *cannot* run any SQL statements in this mode.

Example

The following code snippet shows how to start up a database instance that is down:

```
 OracleDataSource ds = new OracleDataSource();
    Properties prop = new Properties();
    prop.setProperty("user","sys");
    prop.setProperty("password","manager");
    prop.setProperty("internal_logon","sysdba");
    prop.setProperty("prelim_auth","true");
    ds.setConnectionProperties(prop);
    ds.setURL("jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=XYZ.com)
(PORT=5221))"
+ "(CONNECT_DATA=(SERVICE_NAME=rdbms.devplmt.XYZ.com)))");
    OracleConnection conn = (OracleConnection)ds.getConnection();
    conn.startup(OracleConnection.DatabaseStartupMode.NO_RESTRICTION);
    conn.close();
```

> **Note:**
>
> The `startup` method will start up the database using the server parameter file. Oracle JDBC drivers do *not* support database startup using the client parameter file.

39.2.1 Database Startup Options

The `startup` method takes a parameter that specifies the database startup option.

The following table lists the supported database startup options. These options are defined in the `oracle.jdbc.OracleConnection.DatabaseStartupMode` class.

Table 39-1 Supported Database Startup Options

Option	Description
`FORCE`	Shuts down the current instance of the database, if any, in the `ABORT` mode, before starting a new instance.
`NO_RESTRICTION`	Starts up the database with no restrictions.
`RESTRICT`	Starts up the database and allows database access only to users with both the `CREATE SESSION` and `RESTRICTED SESSION` privileges, typically, the DBA.

The `startup` method only starts up a database instance. It neither mounts it nor opens it. For mounting and opening the database instance, you have to reconnect as `SYSDBA` or `SYSOPER`, without the `PRELIM_AUTH` mode.

Example

The following code snippet shows how to mount and open a database instance:

```
OracleDataSource ds1 = new OracleDataSource();
Properties prop1 = new Properties();
prop1.setProperty("user","sys");
prop1.setProperty("password","manager");
prop1.setProperty("internal_logon","sysdba");
ds1.setConnectionProperties(prop1);
ds1.setURL(DB_URL);
OracleConnection conn1 = (OracleConnection)ds1.getConnection();
Statement stmt = conn1.createStatement();
stmt.executeUpdate("ALTER DATABASE MOUNT");
stmt.executeUpdate("ALTER DATABASE OPEN");
```

39.3 Using the shutdown Method

The `shutdown` method enables you to shut down an Oracle Database instance. To use this method, you must be connected to the database as a `SYSDBA` or `SYSOPER`.

Example

The following code snippet shows how to shut down a database instance:

```
    OracleDataSource ds2 = new OracleDataSource();
...
    OracleConnection conn2 = (OracleConnection)ds2.getConnection();
    conn2.shutdown(OracleConnection.DatabaseShutdownMode.IMMEDIATE);
    Statement stmt1 = conn2.createStatement();
    stmt1.executeUpdate("ALTER DATABASE CLOSE NORMAL");
    stmt1.executeUpdate("ALTER DATABASE DISMOUNT");
    stmt1.close();
    conn2.shutdown(OracleConnection.DatabaseShutdownMode.FINAL);
    conn2.close();
```

39.3.1 Database Shutdown Options

Like the `startup` method, the `shutdown` method also takes a parameter. In this case, the parameter specifies the database shutdown option. Table 39-2 lists the supported database shutdown options. These options are defined in the `oracle.jdbc.OracleConnection.DatabaseShutdownMode` class.

Table 39-2 Supported Database Shutdown Options

Option	Description
`ABORT`	Does not wait for current calls to complete or users to disconnect from the database.
`CONNECT`	Refuses any new connection and waits for existing connection to end.
`FINAL`	Shuts down the database.
`IMMEDIATE`	Does not wait for current calls to complete or users to disconnect from the database.
`TRANSACTIONAL`	Refuses new transactions and waits for active transactions to end.
`TRANSACTIONAL_LOCAL`	Refuses new local transactions and waits for active local transactions to end.

For shutdown options other than `ABORT` and `FINAL`, you must call the `shutdown` method again with the `FINAL` option to actually shut down the database.

> **Note:**
>
> The `shutdown(DatabaseShutdownMode.FINAL)` method must be preceded by another call to the `shutdown` method with one of the following options: `CONNECT`, `TRANSACTIONAL`, `TRANSACTIONAL_LOCAL`, or `IMMEDIATE`. Otherwise, the call hangs.

39.3.2 Standard Database Shutdown Process

A standard way to shut down the database is as follows:

1. Initiate shutdown by prohibiting further connections or transactions in the database. The shut down option can be either `CONNECT`, `TRANSACTIONAL`, `TRANSACTIONAL_LOCAL`, or `IMMEDIATE`.
2. Dismount and close the database by calling the appropriate `ALTER DATABASE` command.
3. Finish shutdown using the `FINAL` option.

In special circumstances to shut down the database as fast as possible, the `ABORT` option can be used. This is the equivalent to `SHUTDOWN ABORT` in SQL*Plus.

39.4 A Complete Example

Example 39-1 illustrates the use of the `startup` and `shutdown` methods.

Example 39-1 Database Startup and Shutdown

```
import java.sql.Statement;
import java.util.Properties;
import oracle.jdbc.OracleConnection;
import oracle.jdbc.pool.OracleDataSource;
/**
 * To logon as sysdba, you need to create a password file for user "sys":
 *   orapwd file=/path/orapw password=password entries=300
 * and add the following setting in init.ora:
 *   REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
 * then restart the database.
 */
public class DBStartup
{
  static final String DB_URL = "jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)
(HOST=localhost)(PORT=5221))"
+ "(CONNECT_DATA=(SERVICE_NAME=rdbms.devplmt.XYZ.com)))";

  public static void main(String[] argv) throws Exception
  {
// Starting up the database:
    OracleDataSource ds = new OracleDataSource();
    Properties prop = new Properties();
    prop.setProperty("user","sys");
    prop.setProperty("password","manager");
    prop.setProperty("internal_logon","sysdba");
    prop.setProperty("prelim_auth","true");
    ds.setConnectionProperties(prop);
```

```
    ds.setURL(DB_URL);
    OracleConnection conn = (OracleConnection)ds.getConnection();
    conn.startup(OracleConnection.DatabaseStartupMode.NO_RESTRICTION);
    conn.close();

// Mounting and opening the database
    OracleDataSource ds1 = new OracleDataSource();
    Properties prop1 = new Properties();
    prop1.setProperty("user","sys");
    prop1.setProperty("password","manager");
    prop1.setProperty("internal_logon","sysdba");
    ds1.setConnectionProperties(prop1);
    ds1.setURL(DB_URL);
    OracleConnection conn1 = (OracleConnection)ds1.getConnection();
    Statement stmt = conn1.createStatement();
    stmt.executeUpdate("ALTER DATABASE MOUNT");
    stmt.executeUpdate("ALTER DATABASE OPEN");
    stmt.close();
    conn1.close();

// Shutting down the database
    OracleDataSource ds2 = new OracleDataSource();
    Properties prop = new Properties();
    prop.setProperty("user","sys");
    prop.setProperty("password","manager");
    prop.setProperty("internal_logon","sysdba");
    ds2.setConnectionProperties(prop);
    ds2.setURL(DB_URL);
    OracleConnection conn2 = (OracleConnection)ds2.getConnection();
    conn2.shutdown(OracleConnection.DatabaseShutdownMode.IMMEDIATE);
    Statement stmt1 = conn2.createStatement();
    stmt1.executeUpdate("ALTER DATABASE CLOSE NORMAL");
    stmt1.executeUpdate("ALTER DATABASE DISMOUNT");
    stmt1.close();
    conn2.shutdown(OracleConnection.DatabaseShutdownMode.FINAL);
    conn2.close();
  }
}
```