



## Database LogOn: Native, ODBC and DOR

\*\*\*\*\*

### Native Database Connections

There are two 'levels' with which PCRE logs on and processes the jobs from the native databases.

1. Engine level ( PCREEngine class)
2. Job Level (PCREJob class)

### Engine Level

At the engine level, the engine has to log on to the server every time it needs to create a report. The logOnInfo structure must be explicitly populated as shown in the example below. The Engine level is helpful when there is a requirement to run reports that need to access different databases on different servers or machines. However, every time there is a need to have different information related to the database, it must be set explicitly in the logOninfo structure. In this case use LogOnServer(), to log on to the server, process the required job and then use LogOffserver() at the engine level as shown in the example below.

### Code Sample (Engine Level)

```
Client = new PCREApplicationClient("benchmark1");
Client.Connect();
PCREEngine engine = client.OpenEngine();
PCREJob job = engine.OpenJob("C:\\xxx.rpt");

PCRELogOnInfo logOnInfo = new PCRELogOnInfo();
logOnInfo.ServerName = "MSSQLTEST"; // server name
logOnInfo.DatabaseName = "xxxx"; // database used
logOnInfo.UserID = "dummyuser"; // user name
logOnInfo.Password = "12345"; // password

// logon to the server here
boolean logOnStatus = engine.LogOnServer ( "p2ssql", logOnInfo );

if ( logOnStatus )
    System.out.println ( "Log on succeeded" );
else
    System.out.println ( "Log on failed" );

job.OutputToPDF("C:\\xxx.pdf");

job.Start();

job.Stop();

// logoff the server here
boolean logOffStatus = engine.LogOffServer ( "p2ssql.dll", logOnInfo );
if ( logOffStatus )
    System.out.println ( "Log off succeeded" );
else
    System.out.println ( "Log off failed" );

engine.close();
```



FAQ# 25-09100103  
Modified 09/10/01

## Job Level

At the job level the information of each server is obtained using GetNthTableLogOnInfo( ) method. In this case all information from the database for the respective Job is obtained to make the required logon. Hence different databases and server information is obtained as required by the respective Jobs. However, there is only a need to explicitly specify the password in logOnInfo as shown in the example given below. SetNthTableLogOnInfo() needs to be used to set the logOnInfo at the Job level.

Note: At the operational level and as far as the report generation is concerned, there is no difference between these two methods and besides the differences discussed above there are no advantages or disadvantages between the two approaches. Obviously both will be generating similar reports.

### Code Sample (Job Level)

```
Client = new PCREApplicationClient("benchmark1");
Client.Connect();
PCREEngine engine = client.OpenEngine();
PCREJob job = engine.OpenJob("C:\\xxx.rpt");

// logon to database
PCRELogOnInfo logonInfo;

// get logon info from .rpt
logonInfo = job.GetNthTableLogOnInfo(0);

// only information that is required at the Job level
logonInfo.Password = "12345";

// set logon info to revised values
job.SetNthTableLogOnInfo(0, logonInfo, true);

job.OutputToPDF("C:\\xxx.pdf");

job.Start();

job.Stop();

engine.close();
```

## ODBC Database Connections

When connecting to the data source via an ODBC connection it is important the a System DSN entry be made in the ODBC control panel for every data source used in the main report as well as those used in subreports. The sample code above is the same for ODBC connections but you must replace the native .dll with the ODBC .dll.

## Data Object Reporting (DOR) alternative to standard database connections.

As an alternative to an ODBC or Native connection to the database please consider Parallel Crystal's DOR feature. PCRE includes a feature called Data Object Reporting (DOR) for acquiring a report's data from a programmable object, rather than a database.



**FAQ# 25-09100103**

Modified 09/10/01

DOR uses the JDBC APIs (with our Java library) or ADO APIs (with our ActiveX/COM library) to "push" result set data from the middle tier server into the report server. As soon as the report completes, the copy of the data on the report server is deleted automatically.

This feature is proving to be popular with customers who want to maximize the role of Web Application Servers as the "traffic cop" controlling all aspects of their middle tier servers.

Please see the following links for additional information and samples on DOR:

[http://www.dynamalivery.com/news/news.html#2\\_6release](http://www.dynamalivery.com/news/news.html#2_6release)

<http://www.dynamalivery.com/products/parallecrystal/parallecrystal.doc.html>

<http://www.dynamalivery.com/customerservice/samples/samplecode.html>

<http://www.dynamalivery.com/customerservice/faq/DORCreation.html>