

Crystal Reports

Oracle Stored Procedures

Overview

This document contains information about which database drivers support Oracle stored procedures for Crystal Reports XI Release 2, Crystal Reports XI, Crystal Reports 10, and Crystal Reports 9.. Also, this document contains samples of how to report off an Oracle stored procedure, including a list of minimum requirements.

NOTE

A native or ODBC connection is recommended for reporting off Oracle stored procedures for full functionality to be available.

Contents

INTRODUCTION	2
DATABASE DRIVERS THAT SUPPORT ORACLE STORED PROCEDURES	2
<i>Crystal Reports 9</i>	<i>2</i>
<i>Crystal Reports 10</i>	<i>3</i>
<i>Crystal Reports XI & XI Release 2</i>	<i>2</i>
REQUIREMENTS OF ORACLE STORED PROCEDURES	4
<i>Strongly Bound REF CURSORS versus Weakly Bound REF CURSORS</i>	<i>5</i>
CREATING AN ORACLE STORED PROCEDURE	5
<i>Sample A: Use with ODBC and native connections in Crystal Reports</i>	<i>5</i>
<i>Sample B: Weakly Bound REF CURSOR</i>	<i>6</i>
<i>Sample C: Stored procedure created within a package</i>	<i>7</i>
EXECUTING AN ORACLE STORED PROCEDURE IN SQL*PLUS	9
CREATING A CRYSTAL REPORT OFF AN ORACLE STORED PROCEDURE	9
FINDING MORE INFORMATION	10

Introduction

This article provides information for Oracle stored procedures and Crystal Reports (CR). You will find detailed information on the following:

- Crystal Reports 9, 10, XI, and XI Release 2 database drivers that support Oracle stored procedures
- Requirements for reporting from Oracle stored procedures
- Creating Oracle stored procedures
- Executing Oracle stored procedures in SQL*Plus to troubleshoot

Database Drivers that Support Oracle Stored Procedures

Crystal Reports XI & XI Release 2

For Crystal Reports XI and XI Release 2, refer to **Table C: Database drivers and Oracle**, to determine which database drivers can report off stored procedures in the specified versions of Oracle.

Table B: Database drivers and Oracle client version

Database Drivers (32-bit only)	Oracle 8.05	Oracle 8.0.6	Oracle 8.1.7	Oracle 9.2.x	Oracle 10.x
Native Oracle Driver (Crdb_oracle.dll)	No	No	No	Yes	Yes
CR Oracle 5.0/5.2 ODBC Driver	No	No	Yes	Yes	Yes
Oracle ODBC Driver	No	No	Yes	Yes	Yes

Crystal Reports 10

For Crystal Reports 10, refer to **Table B: Database drivers and Oracle**, to determine which database drivers can report off stored procedures in the specified versions of Oracle.

Table B: Database drivers and Oracle client version

Database Drivers (32-bit only)	Oracle 7.x	Oracle 8.0.5	Oracle 8.0.6	Oracle 8.1.7	Oracle 9.x
Native Oracle Driver (Crdb_oracle.dll)	No	No	No	Yes	Yes
CR Oracle 4.20 ODBC Driver	No	No	No	Yes	Yes
Oracle ODBC Driver	No	No	No	Yes	Yes

NOTE

Only the most recent version of the Oracle 8i ODBC driver will be able to work with stored procedures.

Crystal Reports 9

For Crystal Reports 9, refer to **Table A: Database drivers and Oracle**, to determine which database drivers can report off stored procedures in the specified versions of Oracle.

Table A: Database drivers and Oracle client version

Database Drivers (32-bit only)	Oracle 7.x	Oracle 8.0.5	Oracle 8.0.6	Oracle 8.1.x	Oracle 9.x
Native Oracle Driver (Crdb_oracle.dll)	No	No	Yes	Yes	Yes
CR Oracle 4.10 ODBC Driver	No	No	Yes	Yes	Yes
Oracle ODBC Driver	No	No	No	Yes	Yes

NOTE

Only the most recent version of the Oracle 8i ODBC driver are able to work with stored procedures.

Requirements of Oracle Stored Procedures

In order for Crystal Reports to report off Oracle stored procedures, all of the following requirements must be satisfied:

- To use an ODBC connection to access an Oracle stored procedure, you must create a package that defines the REF CURSOR. This REF CURSOR must be strongly bound to a static pre-defined structure (see Strongly Bound REF CURSORS vs. Weakly Bound REF CURSORS). This package must be created separately and before the creation of the stored procedure.
- A native connection to Oracle in Crystal Reports 9 and Crystal Reports 10 can be used to access an Oracle stored procedure that was created within a package and also an Oracle stored procedure that references a weakly bound REF CURSOR
- The stored procedure must have a parameter that is a REF CURSOR type. Crystal Reports uses this parameter to access and define the result set that the stored procedure returns.
- The REF CURSOR parameter must be defined as IN OUT (read/write mode). After the stored procedure has opened and assigned a query to the REF CURSOR, Crystal Reports will perform a FETCH call for every row from the query's result. This is why the parameter must be defined as IN OUT.
- The parameters can only be defined as IN (input) parameters. Crystal Reports is not designed to work with OUT parameters.
- The REF CURSOR variable must be opened and assigned its query within the procedure.
- The stored procedure can only return one record set. The structure of this record set must not change based on parameters.
- The stored procedure cannot call another stored procedure.
- If you are using the CR ODBC driver, verify that the option **Procedure Return Results** is selected as **On** in the ODBC Driver Configuration setup under the **Advanced** tab.
- If you are using the native Oracle driver and using hard-coded date selection within the stored procedure, the date selection must use either a string representation format of YYYY-DD-MM (where the date field = 2004-01-01) or the **To_Date** formula function with the same format specified (where date field = To_Date(2004-01-01', 'YYYY-MM-DD').
- Most importantly, the stored procedure must be able to execute successfully in Oracle's SQL *Plus utility.

If all of these requirements have been met, verify that the database driver that you are using works with that version of Oracle.

Strongly Bound REF CURSORS versus Weakly Bound REF CURSORS

When declaring the REF CURSOR in an Oracle package, the REF CURSOR may be strongly bound or it may be weakly bound. ODBC connections must reference a strongly bound cursor.

A strongly bound REF CURSOR is bound with the same structure that is used by the outputted SELECT statement in the Oracle stored procedure.

```
CREATE OR REPLACE PACKAGE Test_Package
AS TYPE Test_Type IS REF CURSOR RETURN Test_Table%ROWTYPE;
END Test_Package;
/
```

The example code in **green** displays the REF CURSOR strongly bound. In this example, the REF CURSOR is bound with the row structure of the Test_Table table.

A weakly bound REF CURSOR is not assigned a structure in the Oracle package.

```
CREATE OR REPLACE PACKAGE Test_Package
AS TYPE Test_Type IS REF CURSOR;
END Test_Package;
/
```

Although the REF CURSOR is not assigned a structure in the Oracle package, the Oracle stored procedure itself must return a record set with a structure that does not change based on parameters.

Creating an Oracle Stored Procedure

Samples of stored procedures and their steps on how to create them are provided. These are samples of Oracle stored procedures that will return a result set in Crystal Reports. If these samples were executed in the Oracle connection utility, SQL *Plus, they should run successfully.

NOTE	The following stored procedure samples are provided as a tip. Refer to your Oracle documentation for more information on creating stored procedures.
-------------	--

Sample A: Use with ODBC and native connections in Crystal Reports

1. Create a table.

```
CREATE TABLE Test_Table
(ID number(5),
Firstname varchar2(30),
```

```

        Lastname varchar2(30),
        Birthday date);

```

2. Insert values into the table.

```

INSERT INTO Test_Table VALUES
(1, 'Christopher', 'Jones', '01-Nov-70');
INSERT INTO Test_Table VALUES
(2, 'Maria', 'Marshall', '02-Jan-77');
INSERT INTO Test_Table VALUES
(3, 'Jonathan', 'Campbell', '09-Aug-75');
INSERT INTO Test_Table VALUES
(4, 'Julie', 'Gagnon', '23-Dec-72');
INSERT INTO Test_Table VALUES
(5, 'Daemon', 'Thompson', '11-Feb-69');

```

3. Create the Package.

```

CREATE OR REPLACE PACKAGE Test_Package
AS TYPE Test_Type IS REF CURSOR RETURN Test_Table%ROWTYPE;
END Test_Package;
/

```

4. Create the stored procedure.

```

CREATE OR REPLACE PROCEDURE Test_Procedure (
    Test_Cursor IN OUT Test_Package.Test_Type,
    Test_Parameter IN Test_Table.ID%TYPE)
AS
BEGIN
    OPEN Test_Cursor FOR
        SELECT *
        FROM Test_Table
        WHERE Test_Table.ID = Test_Parameter;
END Test_Procedure;
/

```

5. Once the stored procedure is successfully created on your Oracle database, execute the stored procedure.

Sample B: Weakly Bound REF CURSOR

Only use this sample with a native connection to Crystal Reports 9 or later.

1. Create a table.

```

CREATE TABLE Test_Table
(ID number(5),
 Firstname varchar2(30),

```

```

        Lastname varchar2(30),
        Birthday date);

```

2. Insert values into the table.

```

INSERT INTO Test_Table VALUES
(1, 'Christopher', 'Jones', '01-Nov-70');
INSERT INTO Test_Table VALUES
(2, 'Maria', 'Marshall', '02-Jan-77');
INSERT INTO Test_Table VALUES
(3, 'Jonathan', 'Campbell', '09-Aug-75');
INSERT INTO Test_Table VALUES
(4, 'Julie', 'Gagnon', '23-Dec-72');
INSERT INTO Test_Table VALUES
(5, 'Daemon', 'Thompson', '11-Feb-69');

```

3. Create the Package.

```

CREATE OR REPLACE PACKAGE Test_Package
AS TYPE Test_Type IS REF CURSOR;
END Test_Package;
/

```

4. Create the stored procedure.

```

CREATE OR REPLACE PROCEDURE Test_Procedure (
    Test_Cursor IN OUT Test_Package.Test_Type,
    Test_Parameter IN Test_Table.ID%TYPE)
AS
BEGIN
    OPEN Test_Cursor FOR
        SELECT *
        FROM Test_Table
        WHERE Test_Table.ID = Test_Parameter;
END Test_Procedure;
/

```

5. Once the stored procedure is successfully created on your Oracle database, execute the stored procedure.

Sample C: Stored procedure created within a package

Only use this sample with a native connection to Crystal Reports 9 or later.

1. Create a table.

```

CREATE TABLE Test_Table
(ID number(5),

```

```
Firstname varchar2(30),  
Lastname varchar2(30),  
Birthday date);
```

2. Insert values into the table.

```
INSERT INTO Test_Table VALUES  
(1, 'Christopher', 'Jones', '01-Nov-70');  
INSERT INTO Test_Table VALUES  
(2, 'Maria', 'Marshall', '02-Jan-77');  
INSERT INTO Test_Table VALUES  
(3, 'Jonathan', 'Campbell', '09-Aug-75');  
INSERT INTO Test_Table VALUES  
(4, 'Julie', 'Gagnon', '23-Dec-72');  
INSERT INTO Test_Table VALUES  
(5, 'Daemon', 'Thompson', '11-Feb-69');
```

3. Create the Package.

```
CREATE OR REPLACE PACKAGE Test_Package  
AS TYPE Test_Type IS REF CURSOR RETURN Test_Table%ROWTYPE;  
PROCEDURE Test_Procedure (  
    Test_Cursor IN OUT Test_Type,  
    Test_Parameter IN Test_Table.ID%TYPE  
);  
END Test_Package;  
/
```

4. Create the Package Body (With The Code For The Stored Procedure).

```
CREATE OR REPLACE PACKAGE BODY Test_Package  
AS  
    PROCEDURE Test_Procedure (  
        Test_Cursor IN OUT Test_Type,  
        Test_Parameter IN Test_Table.ID%TYPE  
    ) IS  
    BEGIN  
        OPEN Test_Cursor FOR  
            SELECT *  
            FROM Test_Table  
            WHERE Test_Table.ID = Test_Parameter;  
    END Test_Procedure;  
END Test_Package;  
/
```


- Once the stored procedure is successfully created on your Oracle database, execute the stored procedure.

Executing an Oracle Stored Procedure in SQL*Plus

The following script executes the stored procedure samples created earlier in this document. This script runs in Oracle's native connection utility, SQL*Plus. Remember to change the **<parameter value>** to a valid parameter value (a number between 1 and 5).

```
SET SERVEROUTPUT ON

declare
    test_cursor test_package.test_type;
    resultset test_cursor%rowtype;
begin
    test_procedure(test_cursor, <parameter value>);
    if not test_cursor%isopen then
        dbms_output.put_line('the cursor is not open');
    else
        dbms_output.put_line('the cursor is open');
    end if;
    fetch test_cursor into resultset;
    while test_cursor%found loop
        dbms_output.put_line(resultset.ID);
        dbms_output.put_line(resultset.Firstname);
        dbms_output.put_line(resultset.Lastname);
        dbms_output.put_line(resultset.Birthday);
        fetch test_cursor into resultset;
    end loop;
end;

/
```

NOTE

If the script is executing correctly but you do not see any results displayed on the screen, enter SET SERVEROUTPUT ON at the SQL> prompt to display results to the screen.

Creating a Crystal Report off an Oracle Stored Procedure

Here are the steps to create a new Crystal report based on an Oracle stored procedure:

- In Crystal Reports, click the **File** menu, and then click **New**.
- Click **As a Blank Report**, and then click **OK**.
- In the **Database Expert** dialog box, find your data source. Double-click on the data source and you will see a list of the available Schemas, choose the

Schema that you want and then you will see the **Stored Procedures** node. Double-click on that and you will see the list of stored procedures.

4. Move the procedure(s) you want to report off into the **Selected Tables** box. If the stored procedure has a parameter you will be prompted to enter a value at this point.
5. Click **OK** to close the **Enter Parameter Values** dialog box, and then click **OK** to close the **Database Expert**.

You can now design a Crystal report off an Oracle Stored Procedure.

NOTE

If you do not see the Stored Procedures node in the Database Expert do the following. On the **File** menu, click **Options**. On the **Database** tab, under the **Show** section, select the **Stored Procedures** check box.

Finding More Information

For more information regarding the creation of Oracle stored procedures, refer to www.oracle.com.

► www.businessobjects.com

No part of the computer software or this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from Business Objects.

The information in this document is subject to change without notice. Business Objects does not warrant that this document is error free.

This software and documentation is commercial computer software under Federal Acquisition regulations, and is provided only under the Restricted Rights of the Federal Acquisition Regulations applicable to commercial computer software provided at private expense. The use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at 252.227-7013.

The Business Objects product and technology are protected by US patent numbers 5,555,403; 6,247,008; 6,578,027; 6,490,593; and 6,289,352. The Business Objects logo, the Business Objects tagline, BusinessObjects, BusinessObjects Broadcast Agent, BusinessQuery, Crystal Analysis, Crystal Analysis Holos, Crystal Applications, Crystal Enterprise, Crystal Info, Crystal Reports, Rapid Mart, and Web Intelligence are trademarks or registered trademarks of Business Objects SA in the United States and/or other countries. Various product and service names referenced herein may be trademarks of Business Objects SA. All other company, product, or brand names mentioned herein, may be the trademarks of their respective owners. Specifications subject to change without notice. Not responsible for errors or omissions.

Copyright © 2006 Business Objects SA. All rights reserved.