The RPDO2 stored procedure

Data Exporter (v 1.8.2) · https://i2b2.org/docs/the-rpdo2-stored-procedure/

It is possible to invoke the procedure to output a data table from a table definition directly from a SQL client. This is useful primarily for debugging and for developers.

```

```

Running in SQL (by platform)

PostgreSQL

1. **Load the updated script**

```
\i /path/to/rpdo_refcursor.sql
```

2. **Run the procedure**

```
BEGIN;
 -- 'cur' is the default cursor name
 CALL usp_rpdo2(
   p_table_instance_id  := 322,
   p_result_instance_id := NULL,
   p_min_row            := NULL,
   p_max_row            := NULL
 );
 -- retrieve all rows from the cursor
 FETCH ALL FROM cur;
COMMIT;
```

> **Notes:**

> - Because `usp_rpdo2` now uses an `INOUT cur REFCURSOR DEFAULT 'cur'`, you don't need to supply a cursor name.
> - The `FETCH ALL FROM cur` returns every column of your pivoted result set.

Oracle

1. **Load the script**

```
@/path/to/your_oracle_script.sql
```

2. **Bind and call**

```
VAR rc REFCURSOR;
EXEC usp_rpdo2(
 p_table_instance_id  => 322,
 p_result_instance_id => NULL,
 p_min_row            => NULL,
 p_max_row            => NULL,
 p_refcursor          => :rc
);
```

3. **View the cursor**

```
PRINT rc;
```

> **Notes:**
>
> - `VAR rc REFCURSOR` declares a bind variable in SQL*Plus/SQL Developer.
> - `PRINT rc` streams the full result set into your grid or console.

If your SQL client doesn't support PRINT, the following code will output the cursor as a nice table:

```
DECLARE
  rc          SYS_REFCURSOR;
  cur         INTEGER;
  col_cnt     INTEGER;
  desc_tab    DBMS_SQL.DESC_TAB;
  v_val       VARCHAR2(4000);
  header_line VARCHAR2(32767);
  row_line    VARCHAR2(32767);
BEGIN
  -- 1) Call the proc
  usp_rpdo2(
    p_table_instance_id  => 1,
    p_result_instance_id => NULL,
    p_min_row            => NULL,
    p_max_row            => NULL,
    p_refcursor          => rc
  );

  -- 2) Convert to DBMS_SQL cursor
  cur := DBMS_SQL.TO_CURSOR_NUMBER(rc);

  -- 3) Describe & define columns
  DBMS_SQL.DESCRIBE_COLUMNS(cur, col_cnt, desc_tab);
  FOR i IN 1..col_cnt LOOP
    DBMS_SQL.DEFINE_COLUMN(cur, i, v_val, desc_tab(i).col_max_len);
  END LOOP;

  -- 4) Print header row as CSV
  header_line := NULL;
  FOR i IN 1..col_cnt LOOP
    header_line := header_line
      || CASE WHEN i>1 THEN ',' ELSE '' END
      || '"' || desc_tab(i).col_name || '"';
  END LOOP;
  DBMS_OUTPUT.PUT_LINE(header_line);

  -- 5) Fetch & print each row as CSV
  WHILE DBMS_SQL.FETCH_ROWS(cur) > 0 LOOP
    row_line := NULL;
    FOR i IN 1..col_cnt LOOP
```

```
      DBMS_SQL.COLUMN_VALUE(cur, i, v_val);
      row_line := row_line
        || CASE WHEN i>1 THEN ',' ELSE '' END
        || '"' || REPLACE(NVL(v_val,''),'"','""') || '"';
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(row_line);
  END LOOP;

  -- 6) Clean up
  DBMS_SQL.CLOSE_CURSOR(cur);
END;
```

SQL Server

1. **Load the script**

```
:r C:\path\to\rpdo2.sql
```

2. **Execute the procedure**

```
EXEC dbo.usp_rpdo2
 @TABLE_INSTANCE_ID  = 322,
 @RESULT_INSTANCE_ID = NULL,
 @MIN_ROW            = NULL,
 @MAX_ROW            = NULL;
```

3. **View results**

   - By default, the proc's dynamic SQL writes its pivoted output directly to your result grid.
   - **Optionally**, to capture it in a temp table for further querying:

```
CREATE TABLE #my_results (... define columns ...);
INSERT INTO #my_results
EXEC dbo.usp_rpdo2
@TABLE_INSTANCE_ID  = 322,
@RESULT_INSTANCE_ID = NULL,
@MIN_ROW            = NULL,
```

```
@MAX_ROW            = NULL;

SELECT * FROM #my_results;
DROP TABLE #my_results;
```

> **Notes:**
>
> - Local `#` tables created inside a proc aren't accessible afterward, so if you need post-proc querying, use a global temp table (`##...`) or insert the output as shown above.
> - In SSMS, you'll also see the rows immediately in the grid after `EXEC`.