**Q1 Procedure**

Name: spPlayersInsert

Purpose: Insert a player to the player table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_playerid | players.playerid%TYPE | TBC | Input: to insert playerid for new player. If playerid is null, generate a new id using the procedure  Output: return the new generated playerid back to the software |
| sp\_regnumber | players.regnumber%TYPE | TBC | Input: user insert registration number for newly added player |
| sp\_lastname | players.lastname%TYPE | TBC | Input: user insert last name for newly added player |
| sp\_firstname | players.firstname%TYPE | TBC | Input: user insert first name for newly added player |
| sp\_isactive | players.isactive%TYPE | TBC | Input: user insert the active status for newly added player |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -2 | Insert Error | Input data cannot be insert to the table |
| -201 | Insert Existing ID | Duplicated playerid (PK) |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1 – input playerid manually:

DECLARE

inputID players.playerid%TYPE := 7;

exitcode NUMBER;

BEGIN

spPlayersInsert(inputID, 12323, 'Cristiano', 'Ronaldo', 1, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Playerid: ' || inputID);

DBMS\_OUTPUT.PUT\_LINE('Exit Code: ' || exitcode);

END;

Output:

A black text on a white background

Description automatically generated

Output - if playerid is duplicated:

A black text on a white background

Description automatically generated

Execution 2 – playerid automatically generated:

DECLARE

inputID players.playerid%TYPE;

exitcode NUMBER;

BEGIN

spPlayersInsert(inputID, 12323, 'Cristiano', 'Ronaldo', 1, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Playerid: ' || inputID);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:

A number on a white background

Description automatically generated

**Q1 Procedure**

Name: spPlayersUpdate

Purpose: Update a selected player in the player table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_playerid | players.playerid%TYPE | TBC | Input: to select a player to modify |
| sp\_regnumber | players.regnumber%TYPE | TBC | Input: user insert registration number for newly added player |
| sp\_lastname | players.lastname%TYPE | TBC | Input: user insert last name for newly added player |
| sp\_firstname | players.firstname%TYPE | TBC | Input: user insert first name for newly added player |
| sp\_isactive | players.isactive%TYPE | TBC | Input: user insert the active status for newly added player |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -3 | Update Error | No rows are being modified |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE exitcode INT;

BEGIN

spPlayersUpdate(7, 12323, 'Diego', 'Maradona', 0, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

SELECT \* FROM players

WHERE playerid = 7;

Output:





Execution 2 – modifying non-existing data:

DECLARE exitcode INT;

BEGIN

spPlayersUpdate(987666, 55555, 'Ogeid', 'Anodaram', 1, exitcode); -- UPDATE ERROR: -3

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:



**Q1 Procedure**

Name: spPlayersDelete

Purpose: Delete a selected player in the player table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_playerid | players.playerid%TYPE | TBC | Input: to select a player to delete |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -4 | Delete Error | No rows are being deleted |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution – Executing same playerid twice:

DECLARE

exitcode NUMBER;

BEGIN

spPlayersDelete(7, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

spPlayersDelete(7, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:

A black text with black letters

Description automatically generated with medium confidence

**Q1 Procedure**

Name: spPlayersSelect

Purpose: Select a player in the player table to display

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_playerid | players.playerid%TYPE | TBC | Input: to select a player to display |
| sp\_record | players%ROWTYPE | TBC | Output: to store the fetched record for display |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -6 | TOO\_MANY\_ROWS | Too many rows returned |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1 – Select an existing player:

DECLARE

spRecord players%ROWTYPE;

exitcode NUMBER;

BEGIN

spPlayersSelect(7, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('player id: ' || spRecord.playerid);

DBMS\_OUTPUT.PUT\_LINE('player lastname: ' || spRecord.lastname);

DBMS\_OUTPUT.PUT\_LINE('player firstname: ' || spRecord.firstname);

DBMS\_OUTPUT.PUT\_LINE('player regnumber: ' || spRecord.regnumber);

DBMS\_OUTPUT.PUT\_LINE('player isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:

A screenshot of a computer code

Description automatically generated

Execution 2 – Select a non-existing player:

DECLARE

spRecord players%ROWTYPE;

exitcode NUMBER;

BEGIN

spPlayersSelect(987666, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('player id: ' || spRecord.playerid);

DBMS\_OUTPUT.PUT\_LINE('player lastname: ' || spRecord.lastname);

DBMS\_OUTPUT.PUT\_LINE('player firstname: ' || spRecord.firstname);

DBMS\_OUTPUT.PUT\_LINE('player regnumber: ' || spRecord.regnumber);

DBMS\_OUTPUT.PUT\_LINE('player isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:

A screenshot of a computer code

Description automatically generated

**Q1 Procedure**

Name: spTeamsInsert

Purpose: Insert a team to the teams table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_teamid | teams.teamid%TYPE | TBC | Input: to insert teamid for new team. If teamid is null, generate a new id using the procedure  Output: return the new generated teamid back to the software |
| sp\_teamname | teams.teamname%TYPE | TBC | Input: user insert team name for newly added team |
| sp\_isactive | teams.isactive%TYPE | TBC | Input: user insert the active status for newly added team |
| sp\_jerseycolour | teams.jerseycolour%TYPE | TBC | Input: user insert jersey colour for newly added team |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -2 | Insert Error | Input data cannot be insert to the table |
| -201 | Insert Existing ID | Duplicated playerid (PK) |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1 – input teamid manually:

DECLARE

inputID teams.teamid%TYPE := 600;

exitcode NUMBER;

BEGIN

spTeamsInsert(inputID, 'YellowJays', 1, 'Yellow', exitcode);

DBMS\_OUTPUT.PUT\_LINE('Teamid: ' || inputID);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:

A close up of numbers

Description automatically generated

Output - if teamid is duplicated:

A number on a white background

Description automatically generated

Execution 2 – teamid automatically generated:

DECLARE

inputID teams.teamid%TYPE;

exitcode NUMBER;

BEGIN

spTeamsInsert(inputID, 'YellowJays', 1, 'Yellow', exitcode);

DBMS\_OUTPUT.PUT\_LINE('Playerid: ' || inputID);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:

A close up of text

Description automatically generated

**Q1 Procedure**

Name: spTeamsUpdate

Purpose: Update a selected team in the teams table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_teamid | teams.teamid%TYPE | TBC | Input: to insert teamid for new team. If teamid is null, generate a new id using the procedure  Output: return the new generated teamid back to the software |
| sp\_teamname | teams.teamname%TYPE | TBC | Input: user insert team name for newly added team |
| sp\_isactive | teams.isactive%TYPE | TBC | Input: user insert the active status for newly added team |
| sp\_jerseycolour | teams.jerseycolour%TYPE | TBC | Input: user insert jersey colour for newly added team |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -3 | Update Error | No rows are being modified |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE exitcode NUMBER;

BEGIN

spTeamsUpdate(600, 'YellowYays', 1, 'Big Yellow', exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

SELECT \* FROM teams

WHERE teamid = 600;

Output:



A group of black and white words

Description automatically generated

Execution 2 – modifying non-existing data:

DECLARE exitcode NUMBER;

BEGIN

spTeamsUpdate(666, 'BlackPink', 0, 'Black and Pink', exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:



**Q1 Procedure**

Name: spTeamsDelete

Purpose: Delete a selected team in the teams table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_teamid | teams.teamid%TYPE | TBC | Input: to select a team to delete |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -4 | Delete Error | No rows are being deleted |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution – Executing same teamid twice:

DECLARE

exitcode NUMBER;

BEGIN

spTeamsDelete(600, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

spTeamsDelete(600, exitcode); -- NOT EXIST ANY MORE

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:

A black text with black letters

Description automatically generated with medium confidence

**Q1 Procedure**

Name: spTeamsSelect

Purpose: Select a team in the teams table to display

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_teamid | teams.teamid%TYPE | TBC | Input: to select a team to display |
| sp\_record | teams%ROWTYPE | TBC | Output: to store the fetched record for display |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -6 | TOO\_MANY\_ROWS | Too many rows returned |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

Test 1: Select an existing team   
Test 2: Select an non-existing team

DECLARE

spRecord teams%ROWTYPE;

exitcode NUMBER;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('TEST 1:');

spTeamsSelect(600, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Team id: ' || spRecord.teamid);

DBMS\_OUTPUT.PUT\_LINE('Team name: ' || spRecord.teamname);

DBMS\_OUTPUT.PUT\_LINE('Team isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Team jersey colour: ' || spRecord.jerseycolour);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('------------------------------');

DBMS\_OUTPUT.PUT\_LINE('TEST 2:');

spTeamsSelect(666, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Team id: ' || spRecord.teamid);

DBMS\_OUTPUT.PUT\_LINE('Team name: ' || spRecord.teamname);

DBMS\_OUTPUT.PUT\_LINE('Team isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Team jersey colour: ' || spRecord.jerseycolour);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('------------------------------');

END;

Output:

A screenshot of a computer program

Description automatically generated

**Q1 Procedure**

Name: spRostersInsert

Purpose: Insert a roster to the rosters table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_rosterid | rosters.rosterid%TYPE | TBC | Input: to insert rosterid for new roster. If rosterid is null, generate a new id using the procedure  Output: return the new generated rosterid back to the software |
| sp\_playerid | rosters.playerid%TYPE | TBC | Input: user send playerid for new roster insertion |
| sp\_teamid | rosters.teamid%TYPE | TBC | Input: user send teamid for new roster insertion |
| sp\_isactive | rosters.isactive%TYPE | TBC | Input: user insert the active status for newly added roster |
| sp\_jerseynumber | rosters.jerseynumber%TYPE | TBC | Input: user insert the jersey number for newly added player |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -2 | Insert Error | Input data cannot be insert to the table |
| -201 | Insert Existing ID | Duplicated playerid (PK) |
| -1 | Generic error | All other exceptions not handled |
| 0 | Success | Execution successful |
| -5 | NO\_DATA\_FOUND | Playid or teamid does not exist / cannot be found |

Execution 1 – input rosterid manually:

DECLARE

inputID rosters.rosterid%TYPE := 900;

exitcode NUMBER;

BEGIN

spRostersInsert(inputID, 7, 600, 1, 16, exitcode);

DBMS\_OUTPUT.PUT\_LINE('inputID: ' || inputID || ' | ' || 'Exit code: ' || exitcode);

END;

Output:



A close up of a sign

Description automatically generated

Output - if rosterid is duplicated:



Output - if either playid or teamid not exist:



Execution 2 – rosterid automatically generated:

DECLARE

inputID rosters.rosterid%TYPE;

exitcode NUMBER;

BEGIN

spRostersInsert(inputID, 7, 600, 1, 16, exitcode);

DBMS\_OUTPUT.PUT\_LINE('inputID: ' || inputID || ' | ' || 'Exit code: ' || exitcode);

END;

Output:



**Q1 Procedure**

Name: spRostersUpdate

Purpose: Update a selected roster in the rosters table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_rosterid | rosters.rosterid%TYPE | TBC | Input: to insert rosterid for new roster. If rosterid is null, generate a new id using the procedure  Output: return the new generated rosterid back to the software |
| sp\_playerid | rosters.playerid%TYPE | TBC | Input: user send playerid for new roster insertion |
| sp\_teamid | rosters.teamid%TYPE | TBC | Input: user send teamid for new roster insertion |
| sp\_isactive | rosters.isactive%TYPE | TBC | Input: user insert the active status for newly added roster |
| sp\_jerseynumber | rosters.jerseynumber%TYPE | TBC | Input: user insert the jersey number for newly added player |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -3 | Update Error | No rows are being modified |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |
| -5 | NO\_DATA\_FOUND | Playid or teamid does not exist / cannot be found |

Execution 1:

DECLARE exitcode NUMBER;

BEGIN

spRostersUpdate(900, 1026, 626, 1, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:





Execution 2 – modifying non-existing data:

DECLARE exitcode NUMBER;

BEGIN

spRostersUpdate(900, 1026, 600, 1, 99, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

/

Output:



**Q1 Procedure**

Name: spRostersDelete

Purpose: Delete a selected roster in the rosters table

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_rosterid | rosters.rosterid%TYPE | TBC | Input: to select a roster to delete |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -4 | Delete Error | No rows are being deleted |
| -1 | Generic error | All other exceptions not handled |
| 0 | Success | Execution successful |

Execution – Executing same teamid twice:

DECLARE

exitcode NUMBER;

BEGIN

spRostersDelete(900, exitcode); -- TRY TO DELETE TWICE

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END;

Output:

A black text with black letters

Description automatically generated with medium confidence

**Q1 Procedure**

Name: spRostersSelect

Purpose: Select a roster in the rosters table to display

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_rosterid | rosters.rosterid%TYPE | TBC | Input: to select a roster to display |
| sp\_record | rosters%ROWTYPE | TBC | Output: to store the fetched record for display |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -6 | TOO\_MANY\_ROWS | Too many rows returned |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

Test 1: Select an existing roster   
Test 2: Select an non-existing roster

DECLARE

spRecord rosters%ROWTYPE;

exitcode NUMBER;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('TEST 1:');

spRostersSelect(900, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Roster id: ' || spRecord.rosterid);

DBMS\_OUTPUT.PUT\_LINE('Roster playerID: ' || spRecord.playerid);

DBMS\_OUTPUT.PUT\_LINE('Roster teamID: ' || spRecord.teamid);

DBMS\_OUTPUT.PUT\_LINE('Roster isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Roster jersey number: ' || spRecord.jerseynumber);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('------------------------------');

DBMS\_OUTPUT.PUT\_LINE('TEST 2:');

spRostersSelect(901, spRecord, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Roster id: ' || spRecord.rosterid);

DBMS\_OUTPUT.PUT\_LINE('Roster playerID: ' || spRecord.playerid);

DBMS\_OUTPUT.PUT\_LINE('Roster teamID: ' || spRecord.teamid);

DBMS\_OUTPUT.PUT\_LINE('Roster isactive: ' || spRecord.isactive);

DBMS\_OUTPUT.PUT\_LINE('Roster jersey number: ' || spRecord.jerseynumber);

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('------------------------------');

END;

Output:

A screenshot of a computer program

Description automatically generated

**Q2 Procedure**

Name: spPlayersSelectAll

Purpose: Display all players

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csPlayers | Cursor | TBC | Fetch data from players table |
| pRpt | csPlayers%ROWTYPE | TBC | Stores row fetched from csPlayers |

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE exitcode NUMBER;

BEGIN spPlayersSelectAll(exitcode);

END;

Output (Excerpt):

A screenshot of a computer

Description automatically generated

**Q2 Procedure**

Name: spTeamsSelectAll

Purpose: Display all teams

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csTeams | Cursor | TBC | Fetch data from teams table |
| tRpt | csTeams%ROWTYPE | TBC | Stores row fetched from csTeams |

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE exitcode NUMBER;

BEGIN spTeamsSelectAll(exitcode);

END;

Output:

A screenshot of a computer

Description automatically generated

**Q2 Procedure**

Name: spRostersSelectAll

Purpose: Display all rosters

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csRosters | Cursor | TBC | Fetch data from rosters table |
| rRpt | csRosters%ROWTYPE | TBC | Stores row fetched from csRosters |

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE exitcode NUMBER;

BEGIN spRostersSelectAll(exitcode);

END;

Output (Excerpt):

A screenshot of a computer screen

Description automatically generated

**Q3 Procedure**

Name: spPlayersSelectAll (share same name as Q2)

Purpose: Display all players

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csPlayers | SYS\_REFCURSOR | TBC | Output: Fetch data from players table, and pass fetched data back to the caller |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -8 | INVALID\_CURSOR | Cursor is not in a valid state, or cursor has not been opened or has already been closed |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE

exitcode NUMBER;

csPlayers SYS\_REFCURSOR;

pRpt players%ROWTYPE;

found BOOLEAN := FALSE;

BEGIN

spPlayersSelectAll(csPlayers, exitcode);

DBMS\_OUTPUT.PUT\_LINE(

RPAD('PlayerID', 10) ||

RPAD('Reg Number', 15) ||

RPAD('Lastname', 20) ||

RPAD('Firstname', 20) ||

RPAD('Active', 6)

);

LOOP

FETCH csPlayers INTO pRpt;

EXIT WHEN csPlayers%NOTFOUND;

found := TRUE;

DBMS\_OUTPUT.PUT\_LINE(

RPAD(pRpt.playerid, 10) ||

RPAD(pRpt.regnumber, 15) ||

RPAD(pRpt.lastname, 20) ||

RPAD(pRpt.firstname, 20) ||

RPAD(pRpt.isactive, 6)

);

END LOOP;

CLOSE csPlayers;

IF NOT found THEN

exitcode := -5;

DBMS\_OUTPUT.PUT\_LINE('No Data Found. Exit code: ' || exitcode);

END IF;

EXCEPTION

WHEN OTHERS THEN

exitcode := -1;

DBMS\_OUTPUT.PUT\_LINE('Generic error occured. Exit code: ' || exitcode);

END;

Local Variables used in Execution:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csPlayers | SYS\_REFCURSOR | TBC | Receive fetched data from the procedure - spPlayersSelectAll |
| pRpt | players%ROWTYPE | TBC | Stores row fetched from csPlayers |
| exitcode | NUMBER | TBC | Indicates successful (0) or unsuccessful procedure run. |
| found | BOOLEAN | TBC | Indicates if data is found |

Output (Excerpt):

A screenshot of a computer

Description automatically generated

**Q3 Procedure**

Name: spTeamsSelectAll (share same name as Q2)

Purpose: Display all teams

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csTeams | SYS\_REFCURSOR | TBC | Output: Fetch data from teams table, and pass fetched data back to the caller |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -8 | INVALID\_CURSOR | Cursor is not in a valid state, or cursor has not been opened or has already been closed |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE

exitcode NUMBER;

csTeams SYS\_REFCURSOR;

tRpt teams%ROWTYPE;

found BOOLEAN := FALSE;

BEGIN

spTeamsSelectAll(csTeams, exitcode);

DBMS\_OUTPUT.PUT\_LINE(

RPAD('TeamID', 8) ||

RPAD('Team Name', 12) ||

RPAD('Active', 8) ||

RPAD('Jersey Colour', 16)

);

LOOP

FETCH csTeams INTO tRpt;

EXIT WHEN csTeams%NOTFOUND;

found := TRUE;

DBMS\_OUTPUT.PUT\_LINE(

RPAD(tRpt.teamid, 8) ||

RPAD(tRpt.teamname, 12) ||

RPAD(tRpt.isactive, 8) ||

RPAD(tRpt.jerseycolour, 16)

);

END LOOP;

CLOSE csTeams;

IF NOT found THEN

exitcode := -5;

DBMS\_OUTPUT.PUT\_LINE('No Data Found. Exit code: ' || exitcode);

END IF;

EXCEPTION

WHEN OTHERS THEN

exitcode := -1;

DBMS\_OUTPUT.PUT\_LINE('Generic error occured. Exit code: ' || exitcode);

END;

/

Local Variables used in Execution:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csTeams | SYS\_REFCURSOR | TBC | Receive fetched data from the procedure - spTeamsSelectAll |
| tRpt | teams%ROWTYPE | TBC | Stores row fetched from csTeams |
| exitcode | NUMBER | TBC | Indicates successful (0) or unsuccessful procedure run. |
| found | BOOLEAN | TBC | Indicates if data is found |

Output:

A table of sports teams

Description automatically generated with medium confidence

**Q3 Procedure**

Name: spRostersSelectAll (share same name as Q2)

Purpose: Display all rosters

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csRosters | SYS\_REFCURSOR | TBC | Output: Fetch data from rosters table, and pass fetched data back to the caller |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -8 | INVALID\_CURSOR | Cursor is not in a valid state, or cursor has not been opened or has already been closed |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution 1:

DECLARE

exitcode NUMBER;

csRosters SYS\_REFCURSOR;

rRpt rosters%ROWTYPE;

found BOOLEAN := FALSE;

BEGIN

spRostersSelectAll(csRosters, exitcode);

DBMS\_OUTPUT.PUT\_LINE(

RPAD('RosterID', 10)||

RPAD('PlayerID', 10)||

RPAD('TeamID', 8)||

RPAD('Active', 8)||

RPAD('Jersey No.', 10)

);

LOOP

FETCH csRosters INTO rRpt;

EXIT WHEN csRosters%NOTFOUND;

found := TRUE;

DBMS\_OUTPUT.PUT\_LINE(

RPAD(TO\_CHAR(rRpt.rosterid), 10)||

RPAD(TO\_CHAR(rRpt.playerid), 10)||

RPAD(TO\_CHAR(rRpt.teamid), 8)||

RPAD(TO\_CHAR(rRpt.isactive), 8)||

RPAD(TO\_CHAR(rRpt.jerseynumber), 10)

);

END LOOP;

CLOSE csRosters;

IF NOT found THEN

exitcode := -5;

DBMS\_OUTPUT.PUT\_LINE('No Data Found. Exit code: ' || exitcode);

END IF;

EXCEPTION

WHEN OTHERS THEN

exitcode := -1;

DBMS\_OUTPUT.PUT\_LINE('Generic error occured. Exit code: ' || exitcode);

END;

Local Variables used in Execution:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csRosters | SYS\_REFCURSOR | TBC | Receive fetched data from the procedure - spRostersSelectAll |
| rRpt | csRosters%ROWTYPE | TBC | Stores row fetched from csRosters |
| exitcode | NUMBER | TBC | Indicates successful (0) or unsuccessful procedure run. |
| found | BOOLEAN | TBC | Indicates if data is found |

Output (Excerpt):

A screenshot of a computer screen

Description automatically generated

**Q4 View**

Name: vwPlayerRosters

Purpose: A view storing the players for each team in the database

Attributes:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| playerID | Number | 38 | The unique Id of the player |
| pgRegNumber | Varchar2 | 15 | The player’s registration number |
| pFirstName | Varchar2 | 25 | The first name of the player |
| pLastName | Varchar2 | 25 | The last name of the player |
| pIsActive | Number | 38 | If the player is active |
| rPlayerID | Number | 38 | The unique Id of the player |
| rosterID | Number | 38 | The roster identification of the player |
| rTeamID | Number | 38 | The team id that the player belongs to |
| rosterIsActive | Number | 38 | Value of 1 indicates that the roster is active |
| teamName | Varchar2 | 10 | The name of the team the player belongs to |
| tIsActive | Varchar2 | 38 | Value of 1 indicates the team is active |
| tJerseycolour | Varcahr2 | 10 | The jersey color of the player |

Data Sample:

A screenshot of a computer

Description automatically generated

**Q5 Procedure**

Name: spTeamRosterByID

Purpose: Displays a report of the players in team input

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| pTeamId | Number | Default | Input. the team id to be searched for in the query |
| cPlayer | Sys\_refcursor | NOT APPLICABLE | Output. the player data holding teamname, rosterid and fullname  Query:  SELECT  teamName,  rosterID,  pFirstName || ' ' || pLastName AS fullName  FROM vwPlayerRosters  WHERE rTeamID = pTeamID; |
| Exitcode | number | Default | Output. Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

-5 : No Data found related to pTeamId

-1 : generic error, all other exceptions not handled

Execution:

DECLARE

teamID NUMBER := 212;

cTeamRoastPlayer SYS\_REFCURSOR;

teamName VARCHAR2(25);

rosterID NUMBER;

fullName VARCHAR2(50);

exitcode NUMBER;

BEGIN

spTeamRosterByID(teamID, cTeamRoastPlayer, exitcode);

IF exitcode = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Fetching Data...');

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

DBMS\_OUTPUT.PUT\_LINE('Team Name | Roster ID | Full Name');

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

LOOP

FETCH cTeamRoastPlayer

INTO

teamName,

rosterID,

fullName;

EXIT WHEN cTeamRoastPlayer%NOTFOUND;

-- display

DBMS\_OUTPUT.PUT\_LINE(

RPAD(teamName, 9) || ' | ' ||

RPAD(rosterID, 9) || ' | ' ||

RPAD(fullName, 25)

);

END LOOP;

CLOSE cTeamRoastPlayer;

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END IF;

END;

Output:

A screenshot of a computer

Description automatically generated

**Q6 Procedure**

Name: spTeamRosterByName

Purpose: Displays a report of the players in team input

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| tName | Varchar2 | 10 | Input. the team name to be searched for in the query |
| cPlayer | Sys\_refcursor | Default | Output. the player data holding teamname, rosterid and fullname  Query:  SELECT  teamName,  rosterID,  pFirstName || ' ' || pLastName AS fullName  FROM vwPlayerRosters  WHERE rTeamID = pTeamID; |
| Exitcode | number | Default | Output. Indicates successful (0) or unsuccessful procedure run. |

Local Variables: NOT APPLICABLE

Error Codes:

-8 : cursor invalid resulting from a failed Query tName input not existing in database

-1 : generic error, all other exceptions not handled

Execution:

DECLARE

vTeamName VARCHAR2(25);

cTeamRoastPlayer SYS\_REFCURSOR;

cTeamRoast VARCHAR2(25);

rosterID NUMBER;

fullName VARCHAR2(50);

exitcode NUMBER;

BEGIN

vTeamName := '&team\_name'; --get team

spTeamRosterByName(vTeamName, cTeamRoastPlayer, exitcode);

IF exitcode = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

DBMS\_OUTPUT.PUT\_LINE('Team Name | Roster ID | Full Name');

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

LOOP

FETCH cTeamRoastPlayer INTO cTeamRoast, rosterID, fullName;

EXIT WHEN cTeamRoastPlayer%NOTFOUND;

-- Display

DBMS\_OUTPUT.PUT\_LINE(

RPAD(cTeamRoast, 11) || ' | ' ||

RPAD(rosterID, 11) || ' | ' ||

RPAD(fullName, 25)

);

END LOOP;

CLOSE cTeamRoastPlayer;

DBMS\_OUTPUT.PUT\_LINE('----------------------------------------');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Exit code: ' || exitcode);

END IF;

END;

Output:

A screenshot of a computer

Description automatically generated

**Q7 View**

Name: vwTeamsNumPlayers

Purpose: Stores the number of players currently registered on each team

Attributes:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| teamID | TBC | TBC | The team’s unique identifier |
| NumPlayers | TBC | TBC | Count of players registered on the team |

Execution:

SELECT \* FROM vwTeamsNumPlayers;

Output:

A screenshot of a table

Description automatically generated

**Q8 Function**

Name: fncNumPlayersByTeamID

Purpose: Return the number of currently registered players for a selected team, identified by teamID.

Return type: INT

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| team | INT | TBC | Input: select a team to see how many players are registered |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| numOfPlayer | INT | TBC | Input: Count of players registered on the team |

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -5 | NO\_DATA\_FOUND | No data found in the select statement |
| 0 | Success | Execution successful |

Execution 1:

DECLARE

team INT;

res INT;

BEGIN

team := 210;

res:= fncNumPlayersByTeamID(team);

DBMS\_OUTPUT.PUT\_LINE('Number of Player in team # '|| team || ': ' || res);

END;

Output:



**Q9 View**

Name: vwSchedule

Purpose: Shows all games, including the written names for teams and locations.

Attributes:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Table | Alias | Type | Size | Meaning |
| g.gameid | games |  | TBC | TBC | Unique identifier of the games table |
| g.gamenum | games |  | TBC | TBC | Game number |
| g.gamedatetime | games |  | TBC | TBC | Date and time when the game was/will be held |
| g.hometeam | games |  | TBC | TBC | Team ID of the home team |
| t1.teamname | teams | HomeTeamName | TBC | TBC | Team name of the home team |
| g.homescore | games |  | TBC | TBC | Scores obtained by the home team |
| g.visitteam | games |  | TBC | TBC | Team ID of the visiting team |
| t2.teamname | teams | VisitTeamName | TBC | TBC | Team name of the visiting team |
| g.visitscore | games |  | TBC | TBC | Scores obtained by the visiting team |
| g.locationid | games |  | TBC | TBC | Location ID where the game was/will be held |
| locationname | sllocations |  | TBC | TBC | Location name |
| g.isplayed | games |  | TBC | TBC | Indicator of whether the game has been played |
| g.notes | games |  | TBC | TBC | Special notes |

Execution:

SELECT \* FROM vwSchedule;

Output (Excerpt):

**A screenshot of a computer

Description automatically generated**

**Q10 Procedure**

Name: spSchedUpcomingGames

Purpose: Display the games scheduled to be played within the next 'n' days, where 'n' is provided by the user.

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| sp\_ nextGameDay | INT | TBC | Input: the number days indicated by the user |
| exitcode | NUMBER | TBC | Output: Indicates successful (0) or unsuccessful procedure run. |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| found | BOOLEAN | TBC | Indicates if data is found |
| e\_negativeDay | EXCEPTION | TBC | Will be thrown when user input a negative number |

Error Codes:

|  |  |  |
| --- | --- | --- |
| Exit Code | Name | Meaning |
| -5 | NO\_DATA\_FOUND | No games found in the upcoming n days, does not indicate an error |
| -7 | Invalid Input | User sent a negative number where a positive number is expected |
| -1 | Generic error | all other exceptions not handled |
| 0 | Success | Execution successful |

Execution:

DECLARE exitcode INT;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('\*\*\* TEST 1 \*\*\*');

spSchedUpcomingGames(30, exitcode); -- DEFAULT

DBMS\_OUTPUT.PUT\_LINE('Exitcode: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('');

DBMS\_OUTPUT.PUT\_LINE('\*\*\* TEST 2 \*\*\*');

spSchedUpcomingGames(-1,exitcode); -- USER INPUT NEGATIVE DAYS

DBMS\_OUTPUT.PUT\_LINE('Exitcode: ' || exitcode);

DBMS\_OUTPUT.PUT\_LINE('');

DBMS\_OUTPUT.PUT\_LINE('\*\*\* TEST 3 \*\*\*');

spSchedUpcomingGames(0,exitcode); -- USER INPUT 0

DBMS\_OUTPUT.PUT\_LINE('Exitcode: ' || exitcode);

END;

Output – tested on 2 Dec 2023:

A screenshot of a computer program

Description automatically generated

**Q11 Procedure**

Name: spSchedPastGames

Purpose: Displays the games played in the past input days from user’s current date

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| n | Number | Default | Input. The amount of days to be deducted from current date |
| Exitcode | number | Default | Output. Indicates successful (0) or unsuccessful procedure run. |

Local Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| matchCt | Number | Default | Stores the number of games found between (sysdate – n) and sysdate |
| e\_negativeInput | Exception | NOT APPLICABLE | User defined exception, only positive values allowed. Exception raised when input value is below 0 |
| data\_cur | CURSOR | NOT APPLICABLE | Stores the row returned from Query:  SELECT \*  FROM vwSchedule s  WHERE s.gameDateTime BETWEEN TRUNC(SYSDATE) - n AND TRUNC(SYSDATE)  AND s.isplayed = 1; |
| Data\_rec | vwSchedule%ROWTYPE | NOT APPLICABLE | Stores the row fetched from data\_cur; Data type matches vwSchedule attributes |

Error Codes:

-1 : generic error, all other exceptions not handled

-5 : no data found related to games query

-6 : too many rows returned in a query, only 1 row per loop is handled

-7 : invalid input, number of days to deduct from current date less than 0

Execution:

DECLARE exitcode INT;

BEGIN

spSchedPastGames(60, exitcode);

DBMS\_OUTPUT.PUT\_LINE('Exitcode: ' || exitcode);

END;

Output:

A screenshot of a computer

Description automatically generated

**Q12 Table**

Name: tempStandings

Purpose: A table storing each teams calculated value of wins, loses, ties, points, total goals, goals scored against, and goals difference

Attributes:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| Theteamid | Number | 38 | The team’s unique identifier |
| Teamname | Varchar2 | 10 | The team’s name |
| Gp | Number | Default | The total games played by team |
| W | Number | Default | The sum of total games won |
| L | Number | Default | The sum of total games lost |
| T | Number | Default | The sum of total games tied |
| Pts | Number | Default | The sum of total points |
| Gf | Number | Default | The sum of total goals scored |
| Ga | Number | Default | The sum of total goals scored against the team |
| Gd | Number | Default | The difference between goals scored and goals scored against |

Data Sample:

A table with numbers and symbols

Description automatically generated

**Q12 Procedure**

Name: spRunStandings

Purpose: Displays the data stored within tempStandings table

Parameters: Not applicable

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| csStandings | Cursor | NOT APPLICABLE | Stores the row returned from the query:  SELECT \* FROM tempStandings |
| sRpt | ROWTYPE | NOT APPLICABLE | Stores the row fetched from csStandings |
| found | Boolean | NOT APPLICABLE | Value is true indicating query successfully returned a row |

Error Codes: NOT APPLICABLE

Execution:

DECLARE exitcode NUMBER;

BEGIN spRunStandings;

END;

Output:

A table of numbers and names

Description automatically generated

**Q13 Trigger**

Name: trgUpdateTempStandings

Purpose: Automates the update of tempStandings table and executes spRunStandings

Error Codes: NOT APPLICABLE

Execution: UPDATE games SET homescore = 96 WHERE gameid = 1;

Output:

A table of numbers and names

Description automatically generated

**Q14 Procedure**

Name: spMVPperteam

Purpose: Displays a list of top players in each team. Teams included in the list are teams who have scored at least 1 goal.

Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| Exitcode | number | Default | Output. Indicates successful (0) or unsuccessful procedure run. |

Local Variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Size | Meaning |
| mvp\_cur | CURSOR | NOT APPLICABLE | Stores the row returned from the query:  SELECT  p.playerid,  lastname,  firstname,  SUM(numgoals) AS goalCt,  g.teamid,  teamname  FROM players p  JOIN goalscorers g ON p.playerID = g.playerid  JOIN teams t ON g.teamid = t.teamid  HAVING(SUM(numgoals)) || g.teamid IN (  SELECT MAX(goalCt) || teamid  FROM(  SELECT p.playerid,  lastname,  firstname,  SUM(numgoals) AS goalCt,  teamid  FROM players p  JOIN goalscorers g ON p.playerID = g.playerid  GROUP BY  p.playerid,  lastname,  firstname,  teamid  )  GROUP BY teamid  )  GROUP BY  p.playerid,  lastname,  firstname,  g.teamid,  teamname  ORDER BY  goalct DESC,  g.teamID DESC; |
| mRpt | mvp\_cur%ROWTYPE | NOT APPLICABLE | Stores the row fetched from mvp\_cur |
| found | Boolean | NOT APPLICABLE | Value is true indicating query successfully returned a row |

Error Codes:

-8 : cursor invalid resulting from a failed Query stored in mvp\_cur

-6 : too many rows returned in a query, only 1 row per loop is handled

-5 : No Data found related to mvp\_cur

-1 : generic error, all other exceptions not handled

Execution:

DECLARE exitcode NUMBER;

BEGIN spMVPperteam(exitcode);

END;

Output:

A screenshot of a computer

Description automatically generated A white screen with black text

Description automatically generated

**Exit Codes**

Purpose:

Error Codes simplify exception handling of each Procedure and Function

Definition:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Value | Meaning |
| Success | Number | 0 | Procedure or Function executed successfully |
| Generic Error | Number | -1 | All other exceptions not handled |
| Insert Error | Number | -2 | Table row Insertion failed. 0 row counts returned after Insert operation executed |
| Insert Existing ID | Number | -201 | Table row insertion error related to inserting existing primary id |
| Update Error | Number | -3 | Table row Update failed. 0 row counts returned after Update operation executed. |
| Delete Error | Number | -4 | Table row Delete failed. 0 row counts returned after Delete operation executed. |
| No Data Found Error | Number | -5 | No data returned from query |
| Too Many Rows returned | Number | -6 | More than 1 row returned from query |
| Invalid Input | Number | -7 | Input invalid. Validity of input is function/procedure dependent. |
| Invalid Cursor | Number | -8 | Fetching from a cursor that is not open or non existing |