User's Guide for Sports League Procedures, Functions & Triggers

Table of Contents:

Procedure Q1: CRUD Operations for PLAYERS, ROSTER and TEAMS tables

PLAYERS Table

spPlayersInsert

- **Purpose**: Inserts a new player into the PLAYERS table, generating a new ID if not provided.
- **Input Parameters**:

    - `p_player_id` (INTEGER, optional): Player ID. If NULL, a new ID is generated.
    - `p_reg_number` (VARCHAR2): Unique registration number of the player.
    - `p_last_name` (VARCHAR2): Last name of the player.
    - `p_first_name` (VARCHAR2): First name of the player.
    - `p_is_active` (INTEGER): Indicates active (1) or inactive (0) status of the player.
    - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: None directly. Errors indicated through `p_error_code`.
- **Error Codes**:
    - `-2`: Duplicate player ID or registration number.
    - `-3`: Data type/format mismatch.
    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spPlayersInsert(NULL, '12345', 'Doe', 'John', 1, v_error_code);
  IF v_error_code <> 0 THEN
    -- Handle error
  END IF;
END;
```

spPlayersUpdate

- **Purpose**: Updates an existing player's details in the PLAYERS table.
- **Input Parameters**:
    - `p_player_id` (INTEGER): The unique ID of the player to be updated.
    - `p_reg_number` (VARCHAR2): The new registration number for the player.
    - `p_last_name` (VARCHAR2): The new last name for the player.
    - `p_first_name` (VARCHAR2): The new first name for the player.
    - `p_is_active` (INTEGER): Indicates the new active status of the player (1 for active, 0 for inactive).
    - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: The player's record is updated. The success or failure is indicated by `p_error_code`.
- **Error Codes**:
    - `-1`: No player found with the provided ID.
    - `-3`: Data type/format mismatch.
    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spPlayersUpdate(123, '54321', 'Smith', 'Jane', 1, v_error_code);
```

```
  -- Handle error based on v_error_code
END;
```

spPlayersDelete

- **Purpose**: Deletes a player from the PLAYERS table based on their ID.
- **Input Parameters**:
  - `p_player_id` (INTEGER): The ID of the player to be deleted.
  - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: The specified player's record is deleted. The success or failure is indicated by `p_error_code`.
- **Error Codes**:
  - `-1`: No player found with the provided ID.
  - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spPlayersDelete(123, v_error_code);
  -- Handle error based on v_error_code
END;
```

spPlayersSelect

- **Purpose**: Selects and returns details of a player from the PLAYERS table based on their ID.
- **Input Parameters**:
  - `p_player_id` (INTEGER): The ID of the player to be selected.
  - `p_reg_number` (OUT, VARCHAR2): Output parameter for the player's registration number.
  - `p_last_name` (OUT, VARCHAR2): Output parameter for the player's last name.
  - `p_first_name` (OUT, VARCHAR2): Output parameter for the player's first name.
  - `p_is_active` (OUT, INTEGER): Output parameter indicating whether the player is active (1) or not (0).
  - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: Player details are returned in the output parameters. The success or failure is indicated by `p_error_code`.
- **Error Codes**:
  - `-1`: No player found with the provided ID.
  - `-2`: Multiple players found with the provided ID.
  - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_reg_number VARCHAR2(100);
  v_last_name VARCHAR2(100);
```

```
  v_first_name VARCHAR2(100);
  v_is_active INTEGER;
  v_error_code INTEGER;
BEGIN
  spPlayersSelect(123, v_reg_number, v_last_name, v_first_name,
v_is_active, v_error_code);
  -- Handle error and use output parameters
END;
```

spTeamsInsert

- **Purpose**: Inserts a new team into the TEAMS table, generating a new ID if not provided.
- **Input Parameters**:
  - `p_team_id` (INTEGER, optional): Team ID. If NULL, a new ID is generated.
  - `p_team_name` (VARCHAR2): Name of the team.
  - `p_is_active` (INTEGER): Indicates active (1) or inactive (0) status of the team.
  - `p_jersey_colour` (VARCHAR2): Colour of the team's jersey.
  - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: None directly. Errors are indicated through `p_error_code`.
- **Error Codes**:
  - `-2`: Duplicate team ID.
  - `-3`: Data type/format mismatch.
  - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spTeamsInsert(NULL, 'Tigers', 1, 'Orange', v_error_code);
  -- Handle error based on v_error_code
END;
```

ROSTER Table

spRostersInsert

- **Purpose**: Inserts a new roster record into the ROSTERS table.
- **Input Parameters**:
  - `p_roster_id` (INTEGER, optional): Roster ID. If NULL, a new ID is generated.
  - `p_player_id` (INTEGER): Player ID in the roster.
  - `p_team_id` (INTEGER): Team ID in the roster.
  - `p_is_active` (INTEGER): Indicates if the roster is active (1) or not (0).
  - `p_jersey_number` (INTEGER): Jersey number of the player in the roster.
  - `p_error_code` (OUT, INTEGER): Error code indicating operation status.
- **Expected Output**: Inserts a roster record. Errors indicated through `p_error_code`.

- **Error Codes**:
  - `-1`: Validation failed for player or team.
  - `-2`: Duplicate roster ID.
  - `-3`: Data type/format mismatch.
  - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spRostersInsert(NULL, 101, 201, 1, 9, v_error_code);
  -- Handle error based on v_error_code
END;
```

spRostersUpdate

- **Purpose**: Updates an existing roster record in the ROSTERS table.
- **Input Parameters**:
  - `p_roster_id` (INTEGER): ID of the roster to be updated.
  - `p_player_id` (INTEGER): New player ID for the roster.
  - `p_team_id` (INTEGER): New team ID for the roster.
  - `p_is_active` (INTEGER): New active status of the roster.
  - `p_jersey_number` (INTEGER): New jersey number of the player in the roster.
  - `p_error_code` (OUT, INTEGER): Error code indicating operation status.
- **Expected Output**: Updates a roster record. Errors indicated through `p_error_code`.
- **Error Codes**:
  - `-1`: No roster found with provided ID or validation failed.
  - `-3`: Data type/format mismatch.
  - `-4`: Generic/unexpected error.
- **Example**:

```
BEGIN
  spRostersUpdate(10, 101, 201, 0, 8, v_error_code);
  -- Handle error based on v_error_code
END;
```

spRostersDelete

- **Purpose**: Deletes a roster record from the ROSTERS table based on the provided roster ID.
- **Input Parameters**:
  - `p_roster_id` (INTEGER): ID of the roster to be deleted.
  - `p_error_code` (OUT, INTEGER): Error code indicating operation status.
- **Expected Output**: Deletes a roster record. Errors indicated through `p_error_code`.
- **Error Codes**:
  - `-1`: No roster found with the provided ID.

    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spRostersDelete(10, v_error_code);
  -- Handle error based on v_error_code
END;
```

spRostersSelect

- **Purpose**: Selects and returns details of a roster from the ROSTERS table based on the roster ID.
- **Input Parameters**:
    - `p_roster_id` (INTEGER): ID of the roster to be selected.
    - `p_player_id` (OUT, INTEGER): Player ID in the roster.
    - `p_team_id` (OUT, INTEGER): Team ID in the roster.
    - `p_is_active` (OUT, INTEGER): Indicates if the roster is active or not.
    - `p_jersey_number` (OUT, INTEGER): Jersey number of the player in the roster.
    - `p_error_code` (OUT, INTEGER): Error code indicating operation status.
- **Expected Output**: Returns roster details. Errors indicated through `p_error_code`.
- **Error Codes**:
    - `-1`: No roster found with the provided ID.
    - `-2`: Multiple rosters found with the provided ID.
    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_player_id INTEGER;
  v_team_id INTEGER;
  v_is_active INTEGER;
  v_jersey_number INTEGER;
  v_error_code INTEGER;
BEGIN
  spRostersSelect(10, v_player_id, v_team_id, v_is_active,
v_jersey_number, v_error_code);
  -- Handle error and use output parameters
END;
```

TEAMS Table

spTeamsUpdate

- **Purpose**: Updates an existing team's details in the TEAMS table.
- **Input Parameters**:

    - `p_team_id` (INTEGER): The ID of the team to be updated.
    - `p_team_name` (VARCHAR2): The new name for the team.
    - `p_is_active` (INTEGER): The new active status for the team.
    - `p_jersey_colour` (VARCHAR2): The new jersey colour for the team.
    - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: The team's record is updated. Success or failure is indicated by `p_error_code`.
- **Error Codes**:
    - `-1`: No team found with the provided ID.
    - `-3`: Data type/format mismatch.
    - `-4`: Generic/unexpected error.
- **Example**:

```
BEGIN
  spTeamsUpdate(210, 'Lions', 1, 'Blue', v_error_code);
  -- Handle error based on v_error_code
END;
```

spTeamsDelete

- **Purpose**: Deletes a team from the TEAMS table based on the provided team ID.
- **Input Parameters**:
    - `p_team_id` (INTEGER): The ID of the team to be deleted.
    - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: The specified team's record is deleted. Success or failure is indicated by `p_error_code`.
- **Error Codes**:
    - `-1`: No team found with the provided ID.
    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_error_code INTEGER;
BEGIN
  spTeamsDelete(210, v_error_code);
  -- Handle error based on v_error_code
END;
```

spTeamsSelect

- **Purpose**: Selects and returns details of a team from the TEAMS table based on their ID.
- **Input Parameters**:
    - `p_team_id` (INTEGER): The ID of the team to be selected.
    - `p_team_name` (OUT, VARCHAR2): Output parameter for the team's name.
    - `p_is_active` (OUT, INTEGER): Output parameter indicating whether the team is active.
    - `p_jersey_colour` (OUT, VARCHAR2): Output parameter for the team's jersey colour.

    - `p_error_code` (OUT, INTEGER): Output parameter for potential error codes.
- **Expected Output**: Team details are returned in the output parameters. Success or failure is indicated by `p_error_code`.
- **Error Codes**:
    - `-1`: No team found with the provided ID.
    - `-2`: Multiple teams found with the provided ID.
    - `-4`: Generic/unexpected error.
- **Example**:

```
DECLARE
  v_team_name VARCHAR2(100);
  v_is_active INTEGER;
  v_jersey_colour VARCHAR2(100);
  v_error_code INTEGER;
BEGIN
  spTeamsSelect(210, v_team_name, v_is_active, v_jersey_colour,
v_error_code);
  -- Handle error and use output parameters
END;
```

General Functions

fnValidatePlayerAndTeam

- **Purpose**: Validates the existence of a player and a team in the database.
- **Input Parameters**:
    - `p_player_id` (INTEGER): The ID of the player to validate.
    - `p_team_id` (INTEGER): The ID of the team to validate.
- **Returns**: `BOOLEAN` value. `TRUE` if both the player and the team exist, `FALSE` otherwise.
- **Description**:
    - The function checks the existence of a player and a team based on the provided IDs.
    - It performs a count operation on the `PLAYERS` and `TEAMS` tables to verify existence.
    - Ensures that there is only one record for each ID to prevent duplicates.
- **Example**:

```
DECLARE
  player_exists BOOLEAN;
BEGIN
  player_exists := fnValidatePlayerAndTeam(101, 201);
  -- Use player_exists to decide further action
END;
```

Procedure Q2: spPlayersSelectAll, spRostersSelectAll, spTeamsSelectAll

- **Purpose**: Outputs all records from the respective tables (PLAYERS, ROSTERS, TEAMS) using `DBMS_OUTPUT`.
- **Input Parameters**: None.
- **Expected Output**: All records from the respective table displayed in the script window.
- **Potential Errors**:
  - 'Unable to retrieve player data.'
  - 'Unable to retrieve roster data.'
  - 'Unable to retrieve team data.'
- **Example**:

```
BEGIN
  spPlayersSelectAll; -- Replace with respective procedure for ROSTERS
or TEAMS
END;
```

Procedure Q3: spPlayersSelectAll, spRostersSelectAll, spTeamsSelectAll (Cursor Version)

- **Purpose**: Returns a cursor with all records from the respective tables.
- **Input Parameters**: None.
- **Expected Output**: Cursor with all records.
- **Potential Errors**:
  - 'Unable to retrieve player data.'
  - 'Unable to retrieve roster data.'
  - 'Unable to retrieve team data.'
- **Example**:

```
DECLARE
  v_cursor SYS_REFCURSOR;
BEGIN
  spPlayersSelectAll(v_cursor);
  -- Process cursor data
  CLOSE v_cursor;
END;
```

Procedure Q4: vwPlayerRosters (View Creation)

- **Purpose**: Creates a view combining players, rosters, and teams data.
- **Input Parameters**: None (View creation).
- **Expected Output**: A new view `vwPlayerRosters` is created.

Procedure Q5: spTeamRosterByID

- **Purpose**: Displays the team roster for a specified team ID.
- **Input Parameters**: `v_teamid` (NUMBER): Team ID to query.

- **Expected Output**: Roster information of the specified team.
- **Error Codes**: None.
- **Example**:

```
BEGIN
  spteamrosterbyid(123); -- Replace with the desired team ID
END;
```

Procedure Q6: spTeamRosterByName

- **Purpose**: Searches and displays team roster by a partial/full team name.
- **Input Parameters**: `p_team_name` (VARCHAR2): Partial or full team name.
- **Expected Output**: Roster information of teams matching the name.
- **Error Codes**:
  - `1`: No data found.
  - `-3`: Data type/format mismatch.
  - `-4`: Generic/unexpected error.
- **Example**:

```
BEGIN
  spTeamRosterByName('Noobs'); -- Replace with the team name to search
END;
```

Procedure Q7: vwTeamsNumPlayers (View Creation)

- **Purpose**: Creates a view that lists teams along with the number of active players.
- **Input Parameters**: None (View creation).
- **Expected Output**: A new view `vwTeamsNumPlayers` is created.

Function Q8: fncNumPlayersByTeamID

- **Purpose**: Retrieves the number of active players in a team based on team ID.
- **Input Parameters**: `p_team_id` (NUMBER): Team ID.
- **Expected Output**: Number representing count of active players.
- **Error Codes**:
  - `-1`: No team found with given ID.
  - `-4`: Generic/unexpected error.
- **Example**:

```
  DECLARE
    v_num_players NUMBER;
  BEGIN
    v_num_players := fncNumPlayersByTeamID(123); -- Replace with the
desired team ID
```

```
-- Display or use v_num_players
END;
```

Procedure Q9: vwSchedule (View Creation)

- **Purpose**: Creates a view for game schedules.
- **Input Parameters**: None (View creation).
- **Expected Output**: A new view `vwSchedule` is created.

Procedure Q10: spSchedUpcomingGames

- **Purpose**: Retrieves and displays upcoming games within 'n' days.
- **Input Parameters**: `n` (INTEGER): Number of days ahead to retrieve games.
- **Expected Output**: Information about upcoming games.
- **Potential Errors**:
  - 'Error: Invalid number of days.'
    - Solution: Must pass a valid integer.
  - 'Error: Unable to retrieve upcoming games.'
    - Generic error; double-check command.
- **Example**:

```
BEGIN
  spSchedUpcomingGames(7); -- Replace with the number of days ahead
END;
```

Procedure Q11: spSchedPastGames

- **Purpose**: Displays games played in the past 'n' days.
- **Input Parameters**: `n` (NUMBER): Number of past days to query.
- **Expected Output**: Game records from the past 'n' days.
- **Potential Errors**:
  - 'Error: Invalid number of days.'
    - Solution: Must pass a valid integer.
  - 'Error: Unable to retrieve upcoming games.'
    - Generic error; double-check command.
- **Example**:

```
BEGIN
  spSchedPastGames(30); -- Replace with the number of past days
END;
```

Procedure Q12: spRunStandings

- **Purpose**: Updates the `tempStandings` table with current standings data.

- **Input Parameters**: None.
- **Expected Output**: Updated `tempStandings` table.
- **Error Codes**: None.
- **Example**:

```
BEGIN
  spRunStandings;
END;
```

Trigger Q13: trRunStandings

- **Purpose**: Automates the execution of `spRunStandings` on updates in the games table.
- **Input Parameters**: Triggered by update on games table.
- **Expected Output**: Automatic update of standings upon any change in the games table.
- **Error Codes**: None.
- **Example**: User can take advantage of this trigger to run the following query at any time for up-to-date league standings:

```
SELECT * FROM tempStandings
```

Procedure Q14: spGetAllStars

- **Purpose**: Identifies and displays the all-star lineup of players with the most goals for their team in the current season.
- **Input Parameters**: None.
- **Expected Output**: A list of all-star players, with each player's team name, player name, and total goals scored in the season.
- **Potential Errors**:
  - 'Error: Unable to retrieve all star players.'
    - Generic error messages; double-check command.
- **Example**:

```
BEGIN
  spGetAllStars;
END;
```