All Database Columns

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| User Name | Password | First Name | Last Name | Status | Is Playing | Saved Game | Score | Opponent |

* UserName : is the primary key of the table, it has a text data type.
* Password : is a text attribute with no constrains at it.
* FirstName : is the first name of the player and it has a text data type.
* LastName : is the last name of the player and it has a text data type.
* Status : it’s the player status whither is Online or Offline and it has a Boolean data type.
* Is Playing : it’s the player indicator so it represent the current status of the player whither he plays a game or not, it has a Boolean data type.
* SavedGame : it represents the last saved game of the player and it contains of the moves of the two players, the data type is array of characters.
* Score : it represents the score of the player, and the data type of this variable is integer.
* Opponent : it’s represents the opponent of the user when he plays a game to make it easy to save the game with the contributors of it, the data type of the attribute is text.

The Database Table

Players Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| userName | password | firstName | lastName | status | score | isPlaying | gameId |
| PK, Text | Text | Text | Text | Boolean | Integer | Boolean | FK,Integer |

GameLog Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Game\_Id | homePlayer | opponentPlayer | startGameDate | endGameDate | isFinished | savedGame |
| PK,Integer | Text | Text | Timestamp | Timestamp | Boolean | int[9] |

Functions

1. To retrieve all player data :

CREATE OR REPLACE FUNCTION retrieveplayers () RETURNS TABLE ( userplayer text , statusOfPlayer boolean , scoreOfPlayer int) AS $$

DECLARE

BEGIN

RETURN QUERY SELECT username,status,score from players;

END;

$$ LANGUAGE plpgsql;

1. To Check of Player Status

CREATE OR REPLACE FUNCTION isPlayerOnline (NameOfUser text) RETURNS boolean AS $$

DECLARE

statusOfPlayer boolean;

BEGIN

SELECT status INTO statusOfPlayer FROM players WHERE username = NameOfUser;

RETURN statusOfPlayer;

END ;

$$ LANGUAGE plpgsql;

1. To update player score after winning an offline game

CREATE OR REPLACE FUNCTION updatescoreoffline (nameOfUser text) RETURNS VOID AS $$

DECLARE

BEGIN

UPDATE players SET score = score + 5 WHERE username = nameOfUser;

END;

$$ LANGUAGE plpgsql;

1. To update player score after winning an online game

CREATE OR REPLACE FUNCTION updatescoreonline (nameOfUser text) RETURNS VOID AS $$

DECLARE

BEGIN

UPDATE players SET score = score + 20 WHERE username = nameOfUser;

END;

$$ LANGUAGE plpgsql;

1. To Check if player is playing a game

CREATE OR REPLACE FUNCTION isPlaying ( nameOfUser text ) RETURNS BOOLEAN AS $$

DECLARE

isPlayingNow boolean;

BEGIN

SELECT isPlaying INTO isPlayingNow FROM players WHERE username = nameofuser;

RETURN isPlayingNow;

END;

$$ LANGUAGE plpgsql;

1. To create a new user

CREATE OR REPLACE FUNCTION createplayer (nameofuser text ,playerpassword text, firstname text, lastname text) RETURNS VOID AS $$

DECLARE

BEGIN

INSERT INTO players VALUES (nameofuser,playerpassword,firstname,lastname,false,0,false,NULL);

END;

$$ LANGUAGE plpgsql;

1. To add two player to a game

CREATE OR REPLACE FUNCTION createagame (firstuser text , seconduser text) RETURNS int AS $$

DECLARE

Numberofgames int;

BEGIN

SELECT COUNT (game\_id) INTO Numberofgames from gamelog;

Numberofgames = Numberofgames + 1;

INSERT INTO gamelog (game\_id, homePlayer,opponentplayer, startGameDate, isFinished,savedgame) VALUES ( Numberofgames , firstuser,seconduser,now(),false,ARRAY[45,45,45,45,45,45,45,45,45]);

Update players set gameId = Numberofgames where username in (firstuser,seconduser);

RETURN Numberofgames;

END;

$$ LANGUAGE plpgsql;

1. To end game

CREATE OR REPLACE FUNCTION endgame (gamenumber int) RETURNS VOID AS $$

DECLARE

Nowtime timestamp;

BEGIN

Update players set gameId = NULL WHERE gameid = gamenumber;

Nowtime = now();

Update gamelog set endGameDate = nowtime , isFinished = true where game\_id = gamenumber;

End;

$$ LANGUAGE plpgsql;

1. To check in the login

CREATE OR REPLACE FUNCTION loginCheck (nameofuser text ,playerpassword text) RETURNS boolean AS $$

DECLARE

numberOFRows int;

BEGIN

SELECT COUNT (\*) INTO numberOFRows FROM players WHERE username = nameofuser AND password = playerpassword;

IF numberOFRows = 1 THEN RETURN true;

ELSE RETURN false;

END IF;

END;

$$ LANGUAGE plpgsql;

1. To set a game move

CREATE OR REPLACE FUNCTION setmove (fieldNumber int, sign int,gamenumber int) RETURNS void AS $$

DECLARE

BEGIN

UPDATE gamelog SET savedgame[fieldnumber] = sign where game\_id = gamenumber;

END ;

$$ LANGUAGE plpgsql;

1. To get savedGame

CREATE OR REPLACE FUNCTION getSavedData (gamenumber int) RETURNS TABLE (Field1 int,Field2 int,field3 int, field4 int, field5 int, field6 int, field7 int, field8 int, field9 int) AS $$

DECLARE

BEGIN

RETURN QUERY SELECT savedGame[1], savedGame[2], savedGame[3], savedGame[4], savedGame[5], savedGame[6], savedGame[7], savedGame[8], savedGame[9] FROM gamelog WHERE game\_id = gamenumber;

END;

$$ LANGUAGE plpgsql;

1. To check the if userName Available of not

CREATE OR REPLACE FUNCTION checkNewPlayer (nameofuser text) RETURNS boolean AS $$

DECLARE

numberOFRows int;

BEGIN

SELECT COUNT(\*) INTO numberOFRows FROM players WHERE username = nameofuser;

IF numberOfRows = 1 THEN RETURN false;

ELSE RETURN true;

END IF;

END;

$$ LANGUAGE plpgsql;

1. To retrive player’s data

CREATE OR REPLACE FUNCTION retriveplayerdata (nameofuser text)

RETURNS TABLE (firstName text, lastName text , statusOfPlayer boolean , scoreOfPlayer int) AS $$

DECLARE

BEGIN

RETURN QUERY SELECT fname,lname,status,score from players WHERE username = nameofuser;

END;

$$ LANGUAGE plpgsql;

1. To make Player status online

CREATE OR REPLACE FUNCTION makePlayerOnline (nameOfUser text) RETURNS boolean AS $$

DECLARE

BEGIN

UPDATE players SET status = true WHERE username = nameofuser;

RETURN true;

END;

$$ LANGUAGE plpgsql;