

## GENERAL INFORMATION

Query the database by performing:

```
DBQueries db = DBQueries.getInstance()
db.*function*
```

**ResultSet (rs):** A set of rows containing the output of the SQL query. There is a pointer that originally points to just before the first row in the set, so you must call `rs.next()` in order to access the first row of the data.

Useful operations with **ResultSet rs**:

`if/while (rs.next())` — returns true if that next row exists, false otherwise. By accessing the information this way, you already handle calling `rs.next()` to view the first row of data

`rs.getInt(int i)` — returns the integer stored at that row in column `i` (columns begin at 1).

There are similar functions for getting other value types

`rs.getString(String s)` — returns the string stored at that row in the column labeled `s`.

There are similar functions for getting other value types

## MISC FUNCTIONS

[helper] **void nullEmail(String email)** throws `IllegalArgumentException` if email is null

[helper] **void nullPassword(String p)** throws `IllegalArgumentException` if p is null

[helper] **void nullGroup(String group\_id)** throws `IllegalArgumentException` if group\_id is null

[helper] **void validDueDate(String date)** throws an `IllegalArgumentException` if date is null, isn't 10 characters, or if characters 4 and 6 aren't '-'

[helper] **void nullGroupName(String name)** throws an `IllegalArgumentException` if name is null or >20 characters

[helper] **String generateId()** creates a random 20 character string using the English alphabet and 0-9

## USER FUNCTIONS

**int login(String email, String password)** handles login functionality. Returns 0 if email doesn't exist in the database, 1 if password was incorrect, 2 if successful

[helper] **boolean userExists(String email)** returns whether the email exists in the database

**boolean signUp(String email, String password)** handles signing a new user up to the app, returns whether it was successful

**ResultSet userGroups(String email)** returns all the groups a user is in, null if none

**ResultSet userBills(String email)** returns all bill info stored for each bill assigned to the user, null if none

**ResultSet userBillsInGroup(String email, String group\_id)** returns all bill info stored for each bill assigned to the user in the given group, null if none

**boolean deleteAccount(String email)** handles safe deletion of a user's account, and returns whether it was successful

## BILL FUNCTIONS

*[helper]* **boolean billExists(String bill\_id)** returns whether bill\_id is being used in the database

**boolean addBill(String user, String group, String name, double amt, String date, String desc)** handles functionality for adding a bill for a given user\_id and group\_id, returns whether the operation was successful

**boolean deleteBill(String user\_id, String group\_id, String bill\_id)** handles deleting a bill, returns whether the operation was successful

**boolean changeDueDate(String bill\_id, String due\_date)** handles changing the due date for the specified bill, returns whether it was successful

**boolean changeAmount(String bill\_id, double amt)** handles changing the bill's amount value for the specified bill, returns whether it was successful

## GROUP FUNCTIONS

*[helper]* **boolean groupExists(String group\_id)** returns whether the group\_id is being used in the database

**ArrayList<String> groupMembers(String group\_id)** returns an array list of all the email addresses of the users of a given group. The list is empty if the group doesn't exist.

**String groupName(String group\_id)** returns the name associated with the given group\_id, an empty string if the group\_id doesn't exist in the database

*[helper]* **ResultSet groupBills(String group\_id)** returns all bill info stored for each bill in a group, null if none

**boolean createGroup(String user\_id, String group\_name)** handles creating a group for the user with the given name, returns whether it was successful

**int addUserToGroup(String user\_id, String group\_id)** returns 0 if the addition failed, 1 if the user doesn't already exist in the database, 2 if addition was successful

**boolean leaveGroup(String user\_id, String group\_id)** handles a safe removal of a user from a group, returns whether it was successful

**boolean deleteGroup(String group\_id)** handles safe deletion of a group, returns whether it was successful

**int groupParticipation(String group\_id)** returns the number of users in a group, -1 if the group doesn't exist