

Lesson 3 Gosu Queries

This exercise requires that you use **TrainingApp**, **Guidewire Studio**, and a supported web browser. Start **Guidewire Studio for TrainingApp**. Start the server as **Debug** 'Server'.

The default URL for **TrainingApp** is: <http://localhost:8880/ab/ContactManager.do>. Log in to **TrainingApp** as Alice Applegate whose login/password is **aapplegate/gw**.

Exercise 1: Query the assigned user contact



Exercise

Succeed Insurance wants to modify and enhance the **ABContactSummary** class with a query that meets the requirements listed below.

3.1.1 Requirements

Spec 1 Add an additional property to **ABContactSummary**.

- **AssignedUserWorkload** (an integer).

Spec 2 Modify the **loadSummaryData** method that meets the following criteria:

- Query the total number of **Contacts** assigned to the current **AssignedUser** of the displayed contact.
- Set the summary's **AssignedUserWorkload** to this total.
- If the contact's assigned user is null, then set the **AssignedUserWorkload** field to 0.

Spec 3 Modify the **buildConcatenatedSummary** method that meets the following criteria:

- Returns a **String** that contains the value of all five properties as a comma-delimited list.

3.1.2 Tasks

1. **Modify ABContact Summary class.**

- a) Add new **AssignedUserWorkload** property.
- b) Modify **loadSummaryData** method.
- c) Modify **buildConcatenatedSummary** method.

2. **Deploy code changes.**

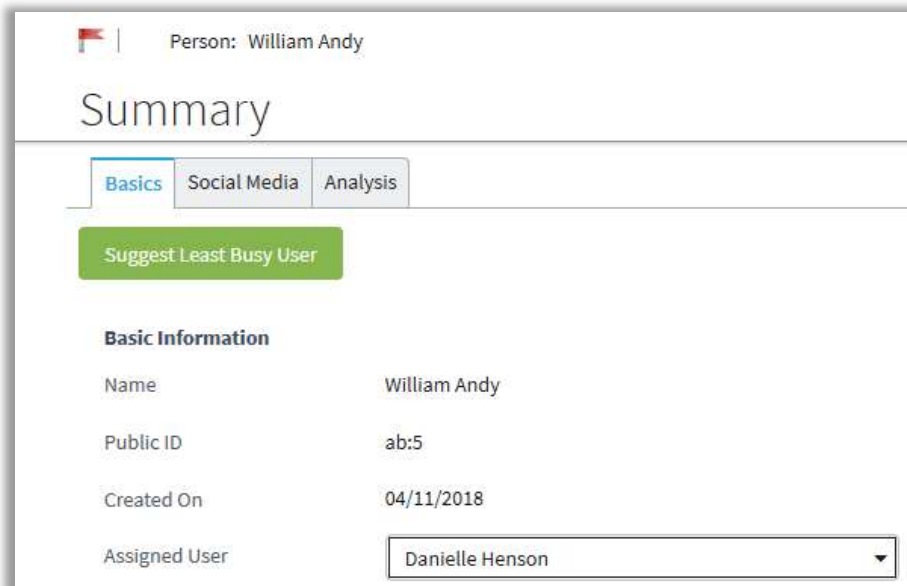
3. **Perform verification steps.**

3.1.3 Verification steps

1. **Assign a User to an ABContact.**

- a) In **TrainingApp**, navigate to the contact that you worked on in the previous exercise, for example: William Andy.

- b) On the **summary** screen, select **Edit** and select an **Assigned User** as Danielle Henson.
- c) Click **Update**.



Person: William Andy

Summary

Basics Social Media Analysis

Suggest Least Busy User

Basic Information

Name	William Andy
Public ID	ab:5
Created On	04/11/2018
Assigned User	Danielle Henson

2. Generate console output for ABContactSummary using Gosu Scratchpad.

- a) In **Studio**, open **Gosu Scratchpad** by clicking **Tools → Gosu Scratchpad**.
- b) Write code that will test the solution:
 - Create a query that references contact William Andy whose **publicID** is **ab:5**.
 - Make sure only one result is retrieved, otherwise throw an exception.
 - Create a new **ABContactSummary**.
 - Initialize the **ExternalID** sequence counter.
 - Execute the **loadSummaryData** method using the given contact.
 - Retrieve and print the output of the **buildConcatenatedSummary** method.

3. Verify the output

- a) Verify that the output in the **Debug Console** has four delimited values with correct values:
 - Sequence number
 - Note: this number can vary based on how many times the sequence utility was executed in the previous exercise
 - Public ID
 - Contact name
 - Number of checking account
 - Assigned User Workload

Solution 1: Query the assigned user contact

1. Modify ABContact Summary class.

- Add new **AssignedUserWorkload** property.
- Modify **loadSummaryData** method.
- Modify **buildConcatenatedSummary** method.

```
package si.ta.classes.entity

uses gw.api.database.Query
uses gw.api.database.Relop
uses gw.api.system.database.SequenceUtil

/**
 * Training lab
 */
class ABContactSummary {

    construct(i: int) {
        ExternalID = i
    }

    // Declare ExternalID property using getter and setter
    var _externalID: int // Don't use Integer because it will initialize as null.

    property get ExternalID(): int {
        return _externalID
    }

    property set ExternalID(externalId: int) {
        if(externalId == 0) {
            _externalID = SequenceUtil.next(1000, "externalID") as int
        } else
            _externalID = externalId
    }

    // Declare properties using shorthand syntax
    var _contactID: String as ContactID
    var _name: String as Name
    var _numCheckingAccounts: int as NumCheckingAccounts
    var _assignedUserWorkload: int as AssignedUserWorkload

    // Create functions

    /**
     * Function that initializes ABContact properties.
     */
    @Param("contact", "Input parameter of type entity.ABContact")
    function loadSummaryData(contact: ABContact): void {
        this._contactID = contact.PublicID
        this._name = contact.DisplayName
        this._numCheckingAccounts = contact.BankAccounts.countWhere(\account ->
            account.AccountType == typekey.BankAccountType.TC_CHECKING)
        // check if the contact actually has an assigned user
        if(contact.AssignedUser != null) {
            var queryObj = Query.make(ABContact)
            queryObj.compare(ABContact#AssignedUser, Relop.Equals, contact.AssignedUser)
            this._assignedUserWorkload = queryObj.select().Count
        } else {
            _assignedUserWorkload = 0
        }
    }
}
```

```

/**
 * Function that builds a comma-delimited list.
 */
@Returns("A comma-delimited list of property values")
function buildConcatenatedSummary(): String {
    return String.format("%s,%s,%s,%s,%s",
        {_externalID, _contactID, _name, _numCheckingAccounts, _assignedUserWorkload})
}
}

```

2. Deploy code changes.

- a) Restart server.

3. Perform verification steps.

```

uses gw.api.database.Query
uses gw.api.database.Relop
uses si.ta.classes.entity.ABContactSummary

var queryObj = Query.make(ABContact)
queryObj.compare(ABContact#PublicID, Relop.Equals, "ab:5")
var targetContact = queryObj.select().AtMostOneRow

var newSummary = new ABContactSummary(0)
newSummary.loadSummaryData(targetContact)
print(newSummary.buildConcatenatedSummary())

```