

INTERMEDIATE

CORE DATA



HANDS-ON CHALLENGES

Intermediate Core Data

Luke Parham

Copyright ©2017 Razeware LLC.

Notice of Rights

All rights reserved. No part of this book or corresponding materials (such as text, images, or source code) may be reproduced or distributed by any means without prior written permission of the copyright owner.

Notice of Liability

This challenge and all corresponding materials (such as source code) are provided on an "as is" basis, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in action of contract, tort or otherwise, arising from, out of or in connection with the software or the use of other dealing in the software.

Trademarks

All trademarks and registered trademarks appearing in this book are the property of their own respective owners.

Challenge #1: Summing with NSExpression

By Luke Parham

In the demo you saw how to fetch the count of objects matching certain predicate. Your challenge is to fill in the label for the **Offering a Deal** section in on the **Filters** screen.

Filling in the Method

To get started, navigate to **FilterViewController.swift** and scroll to the bottom where you'll see `populateDealsCountLabel()` waiting to be implemented.

The first thing to do is create a fetch request with `.dictionaryResultType` for its `resultType` property.

```
let fetchRequest = NSFetchRequest<NSDictionary>(entityName: "Venue")
fetchRequest.resultType = .dictionaryResultType
```

Normally, when you use a dictionary result type you specify an array of properties for which you want values returned.

In this case, you don't want the value of a property that currently exists on Venue types. Instead you're trying to sum together a number from each Venue to be combined and represented as a new property.

This is where `NSExpressionDescription` comes in.

```
let sumExpressionDesc = NSExpressionDescription()
sumExpressionDesc.name = "sumDeals"
sumExpressionDesc.expressionResultType = .integer32AttributeType
```

Here, you create an `NSExpressionDescription` and give it the name **sumDeals**. This is the "property" name you'll provide later. You also specify that you'd like the result as an `Int32`.

Next, add:

```
let specialCountExp = NSEXPRESSION(forKeyPath:
#keyPath(Venue.specialCount))
sumExpressionDesc.expression =
    NSEXPRESSION(forFunction: "sum:", arguments: [specialCountExp])
```

First, you create an NSEXPRESSION for the **specialCount** keyPath. Next you create another expression and tell it you want to use the sum: function and the input to the sum will be the special count of each Venue.

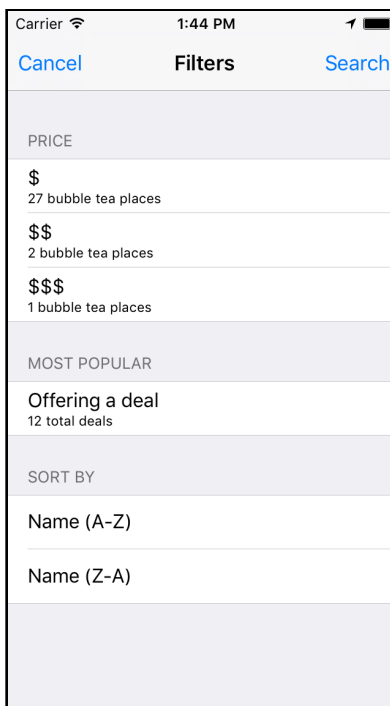
The last thing you need to do is add:

```
fetchRequest.propertiesToFetch = [sumExpressionDesc]

do {
    let results = try CoreDataStack.managedContext.fetch(fetchRequest)
    let resultDict = results.first!
    let numDeals = resultDict["sumDeals"]!
    numDealsLabel.text = "\(numDeals) total deals"
} catch let error as NSError {
    print("Count not fetch \(error), \(error.userInfo)")
}
```

Here, you use sumExpressionDesc as the "property" to be used as the key in the resultant dictionary.

Finally, you use the fetch request you built up and execute it. The result is an array with one value; a dictionary with the key "specialCount" pointing to the sum of all specialCount values.



To learn more about *NSExpression* check out the the [api reference](#).