

Core Data Fundamentals

Performance Tips and Tricks

Brice Wilson
www.BriceWilson.net
[@brice_wilson](https://twitter.com/brice_wilson)



pluralsight 
hardcore developer training

Introduction

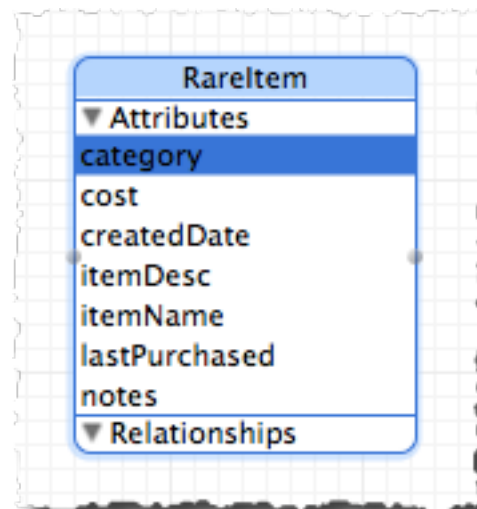
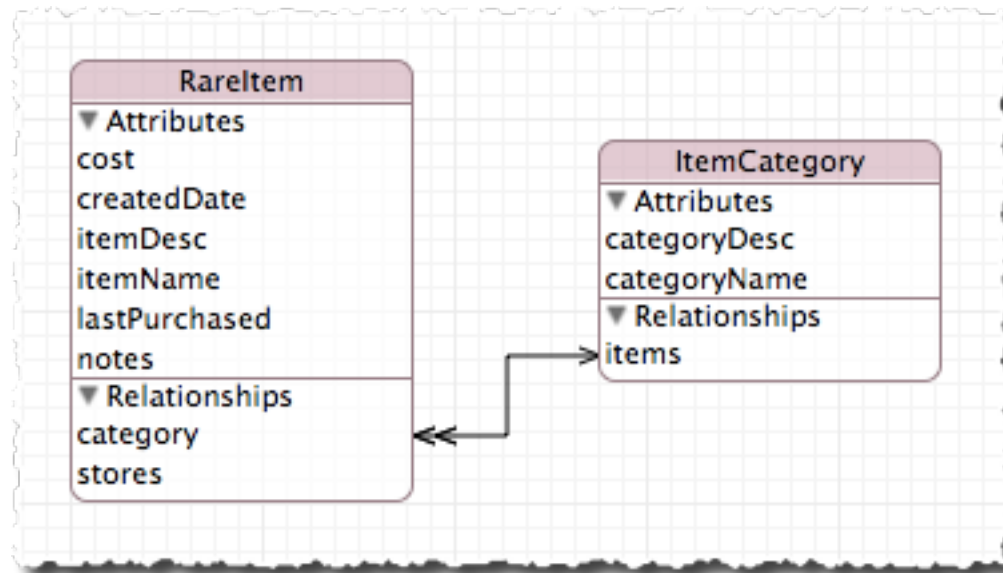
Optimizing the Data Model

Efficient Data Retrieval

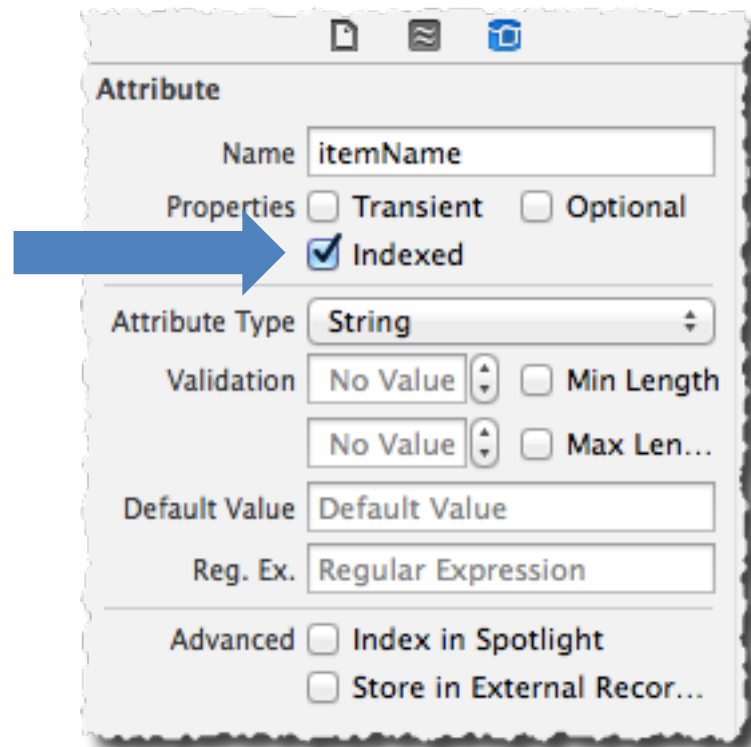
Concurrency Models

Measuring Performance

Normalization



Adding an Index



The image shows a screenshot of a software interface titled "Attribute". It contains several fields and checkboxes for configuring an attribute. A blue arrow points to the "Indexed" checkbox, which is checked. The "Name" field contains "itemName". The "Properties" section has checkboxes for "Transient", "Optional", and "Indexed". The "Attribute Type" is set to "String". The "Validation" section has two rows, each with a "No Value" dropdown and a checkbox for "Min Length" and "Max Len...". The "Default Value" field contains "Default Value". The "Reg. Ex." field contains "Regular Expression". The "Advanced" section has checkboxes for "Index in Spotlight" and "Store in External Recor...".

Attribute

Name

Properties ☐ Transient ☐ Optional ☒ Indexed

Attribute Type

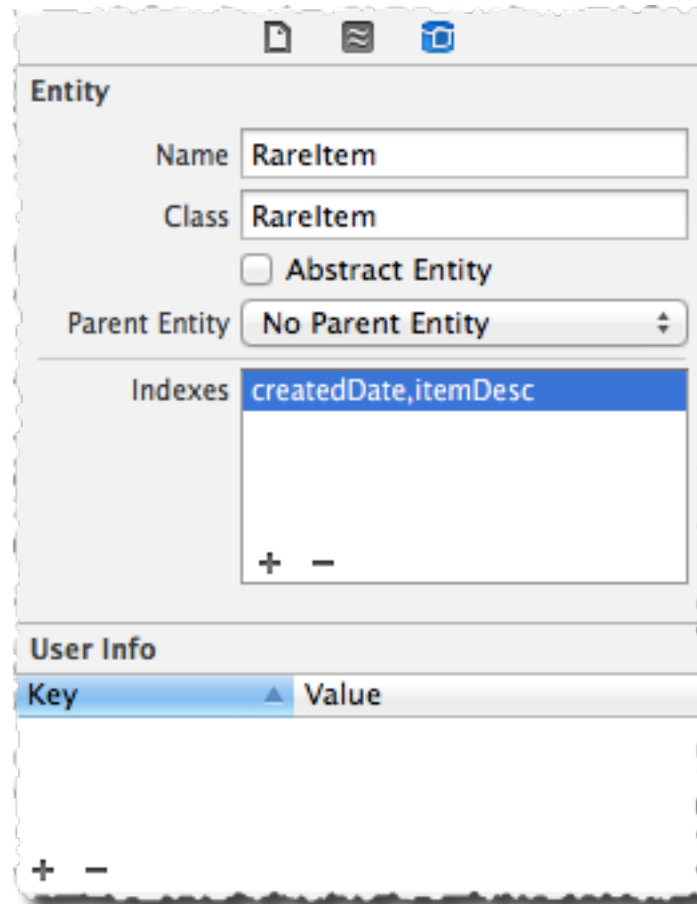
Validation ☐ Min Length ☐ Max Len...

Default Value

Reg. Ex.

Advanced ☐ Index in Spotlight ☐ Store in External Recor...

Adding a Multi-Column Index



The screenshot shows a software interface for configuring an entity. The window has a title bar with standard icons. The main content area is divided into sections. The 'Entity' section contains fields for 'Name' (RareItem), 'Class' (RareItem), an 'Abstract Entity' checkbox, and a 'Parent Entity' dropdown menu set to 'No Parent Entity'. Below this is the 'Indexes' section, which contains a text input field with the value 'createdDate,itemDesc'. This field is highlighted with a blue selection bar. A large blue arrow points from the right side of the image towards this text field. Below the 'Indexes' section is a 'User Info' section with a table header showing 'Key' and 'Value'. At the bottom of the window are expand/collapse icons (+ and -).

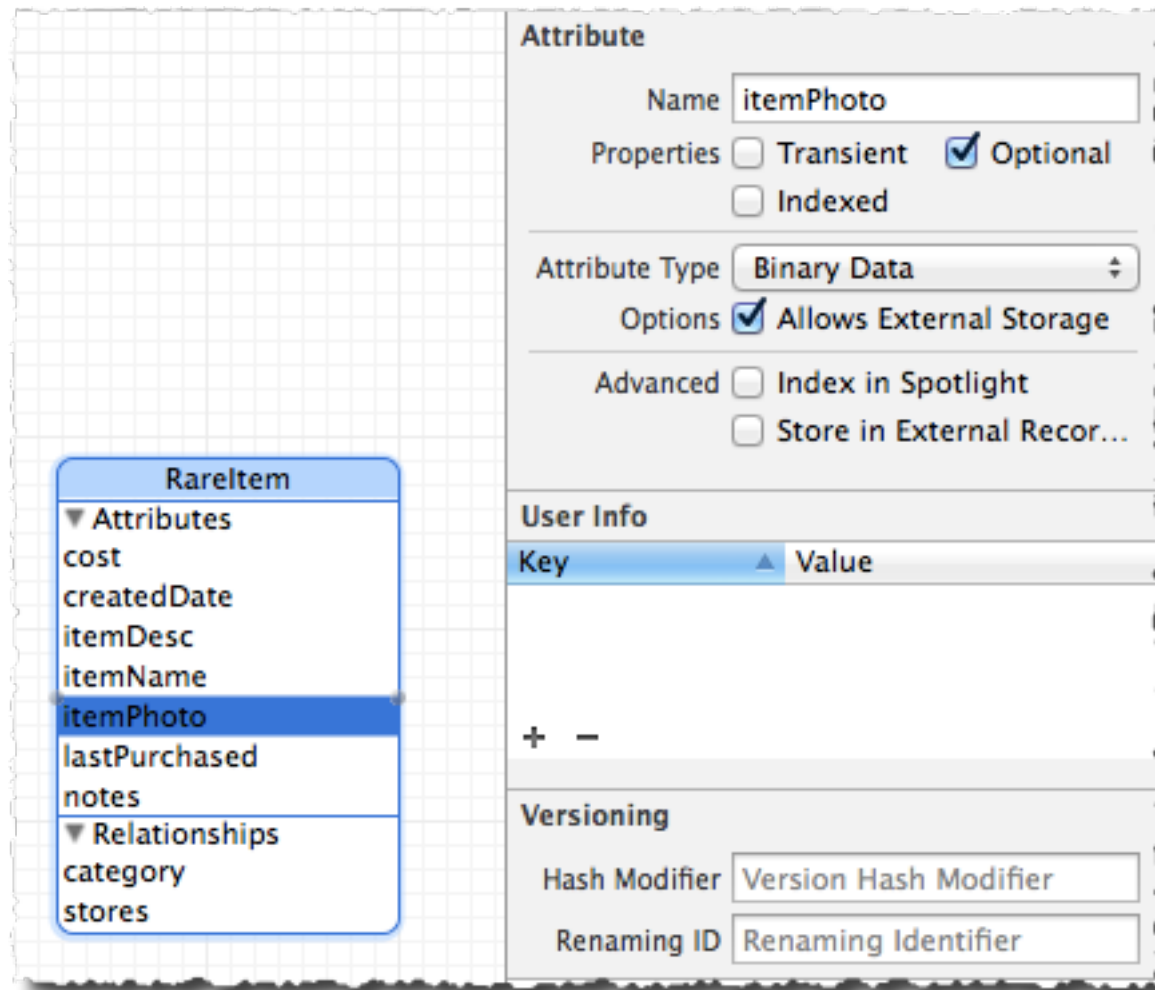
Entity
Name
Class
<input type="checkbox"/> Abstract Entity
Parent Entity
Indexes
createdDate,itemDesc
+
-

User Info
Key
Value

+

-

Storing Binary Data



The image shows the Xcode Core Data editor. On the left, the 'RareItem' entity is selected, and its attributes are listed: cost, createDate, itemDesc, itemName, itemPhoto (highlighted), lastPurchased, and notes. On the right, the 'Attribute' configuration panel for 'itemPhoto' is shown. A blue arrow points to the 'Allows External Storage' checkbox, which is checked.

Attribute

Name:

Properties: ☐ Transient ☒ Optional
☐ Indexed

Attribute Type:

Options: ☒ Allows External Storage

Advanced: ☐ Index in Spotlight
☐ Store in External Recorder

User Info

Key	Value
-----	-------

+ -

Versioning

Hash Modifier:

Renaming ID:

Faulting

RareItem

Baseball Glove

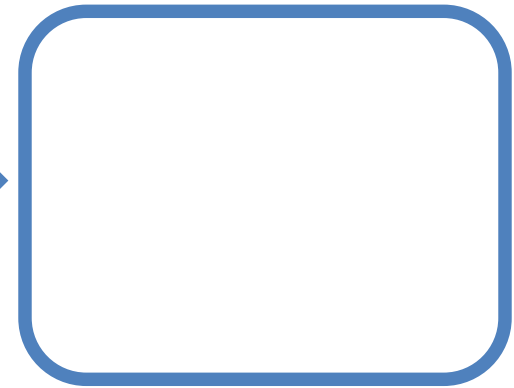
\$20.00

December 1, 2013

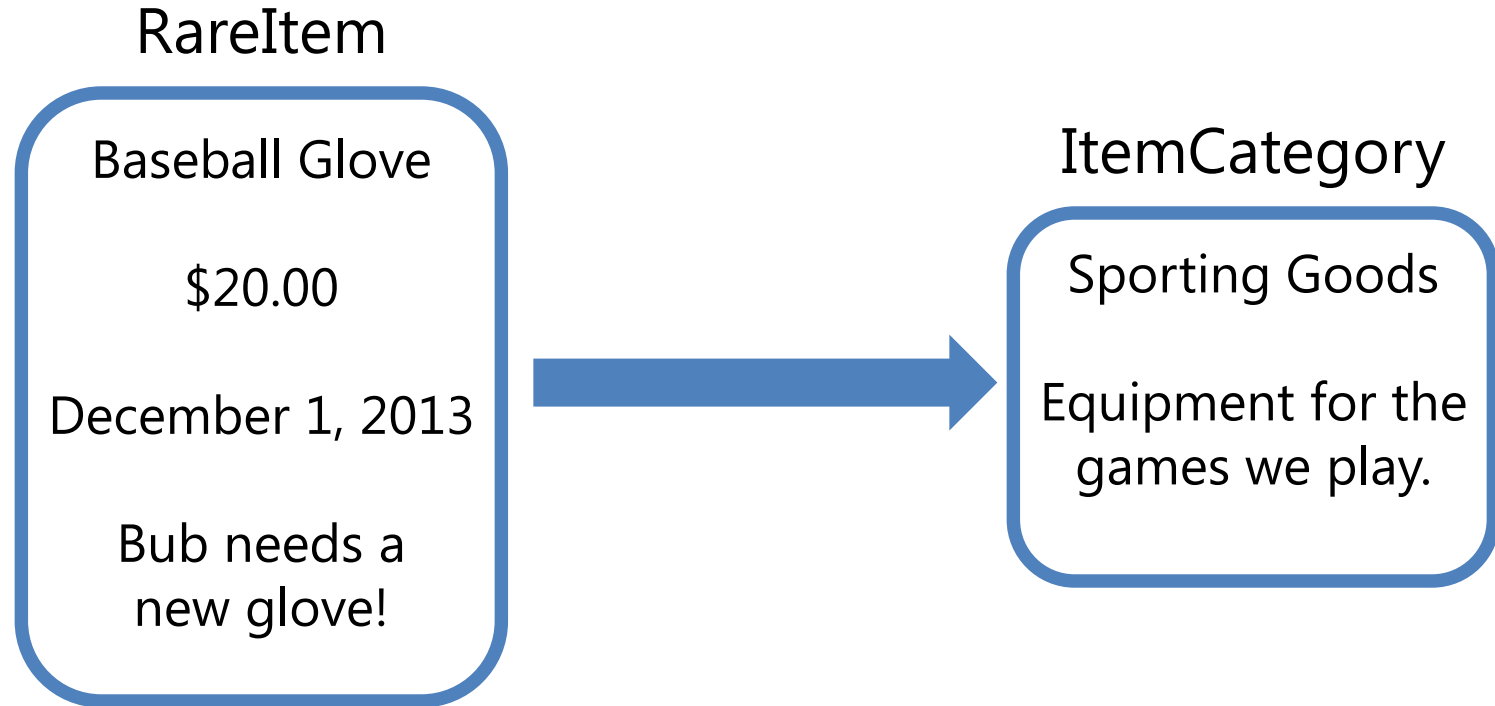
Bub needs a
new glove!



ItemCategory



Faulting



```
ItemCategory *gloveCategory = [baseballGlove category];  
NSLog(@"Category Name: %@", [gloveCategory categoryName]);
```


Refaulting


```
// turn baseballGlove back into a fault to conserve memory  
[context refreshObject:baseballGlove mergeChanges:NO];
```

```
// all managed objects are "forgotten"  
[context reset];
```


Prefetching

```
NSEntityDescription *entity = [NSEntityDescription entityForName:@"RareItem"  
                               inManagedObjectContext:context];
```


```
NSFetchRequest *request = [[NSFetchRequest alloc] init];  
[request setEntity:entity];
```



```
NSMutableArray *prefetchKeys = [NSMutableArray array];  
[prefetchKeys addObject:@"category"];  
[prefetchKeys addObject:@"stores"];
```

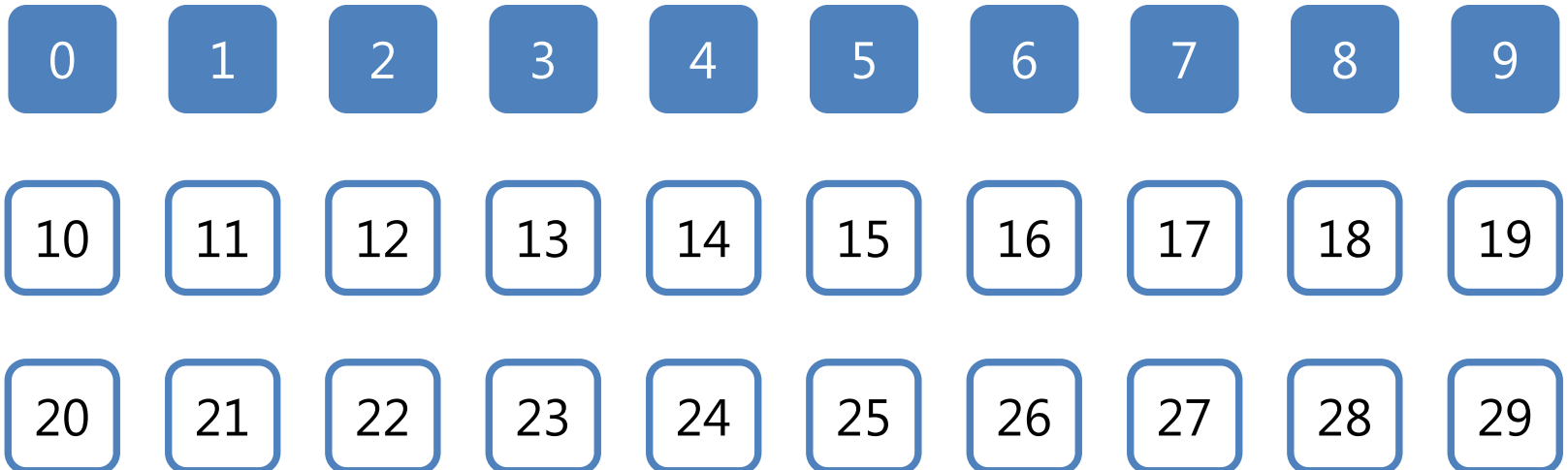
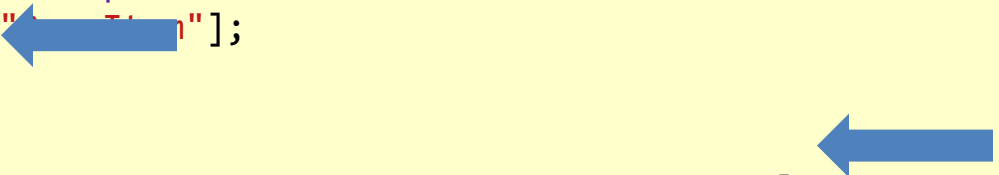


```
[request setRelationshipKeyPathsForPrefetching:prefetchKeys];
```



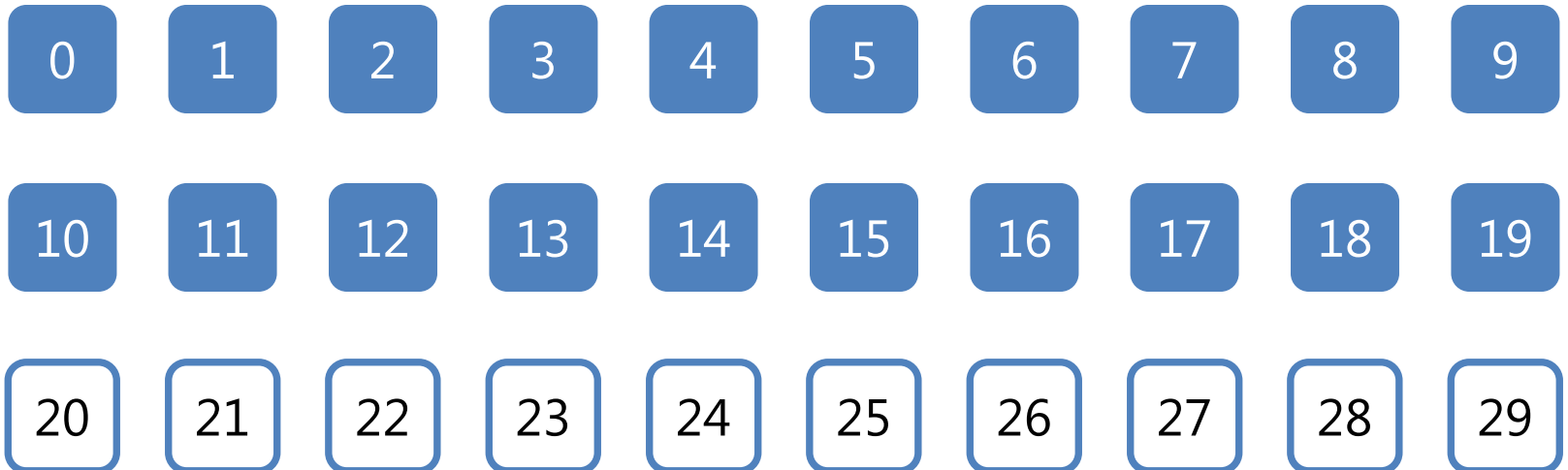

Sip Your Data – Don't Gulp It

```
NSFetchRequest *request = [NSFetchRequest  
    fetchRequestWithEntityName:@"Item"];  
[request setFetchBatchSize:10];  
  
NSArray *allItems = [context executeFetchRequest:request error:&error];  
  
for (RareItem *item in allItems) {  
    NSLog(@"Item Name: %@", [item itemName]);  
}
```




Sip Your Data – Don't Gulp It

```
NSFetchRequest *request = [NSFetchRequest  
    fetchRequestWithEntityName:@"RareItem"];  
[request setFetchBatchSize:10];  
  
NSArray *allItems = [context executeFetchRequest:request error:&error];  
  
for (RareItem *item in allItems) {  
    NSLog(@"Item Name: %@", [item itemName]);  
}
```



Sip Your Data – Don't Gulp It

```
NSFetchRequest *request = [NSFetchRequest  
    fetchRequestWithEntityName:@"RareItem"];  
[request setFetchBatchSize:10];  
  
NSArray *allItems = [context executeFetchRequest:request error:&error];  
  
for (RareItem *item in allItems) {  
    NSLog(@"Item Name: %@", [item itemName]);  
}
```



0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26


27

28

29

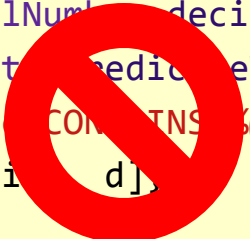
Efficient Text Queries

```
NSString *searchString = @"Garden";  
NSDecimalNumber *d = [NSDecimalNumber decimalNumberWithString:@"30"];  
NSPredicate *pred = [NSPredicate predicateWithFormat:  
    @"itemName CONTAINS %@ AND cost < %@",  
    searchString, d];
```




Efficient Text Queries

```
NSString *searchString = @"Garden";
NSDecimalNumber *d = [NSDecimalNumber decimalNumberWithString:@"30"];
NSPredicate *pred = [NSPredicate predicateWithFormat:
    @"itemName CONTAINS %@ AND cost < %@",
    searchString, d];
```



```
NSString *searchString = @"Garden";
NSDecimalNumber *d = [NSDecimalNumber decimalNumberWithString:@"30"];
NSPredicate *pred = [NSPredicate predicateWithFormat:
    @"cost < %@ AND itemName CONTAINS %@",
    d, searchString];
```



Concurrency Types

NSConfinementConcurrencyType

NSMainQueueConcurrencyType

NSPrivateQueueConcurrencyType

```
NSUInteger type = NSPrivateQueueConcurrencyType;
_context = [[NSManagedObjectContext alloc] initWithConcurrencyType:type];
[_context performBlock:^(
    // work to be performed on the context's queue
)];
```


Summary

Optimize the Data Model

Efficient Data Retrieval

Concurrency Models

Measuring Performance

Thanks for watching!