Realm(.io) of the Mobile Database

(an introduction to Realm)

http://realm.io/

by Martin Grider
@livingtech — http://chesstris.com

Contents of the Realm

- What is Realm?
- Why would you use it?
- Some examples (& code)
- Demo: A trivia app I made with Realm
- Q&A

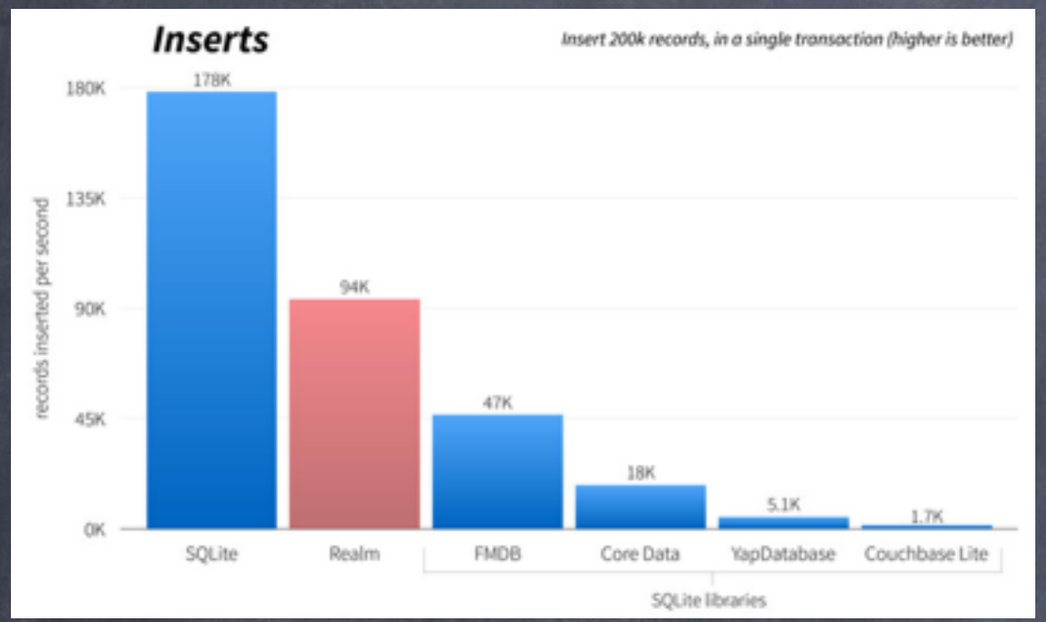
What is Realm?

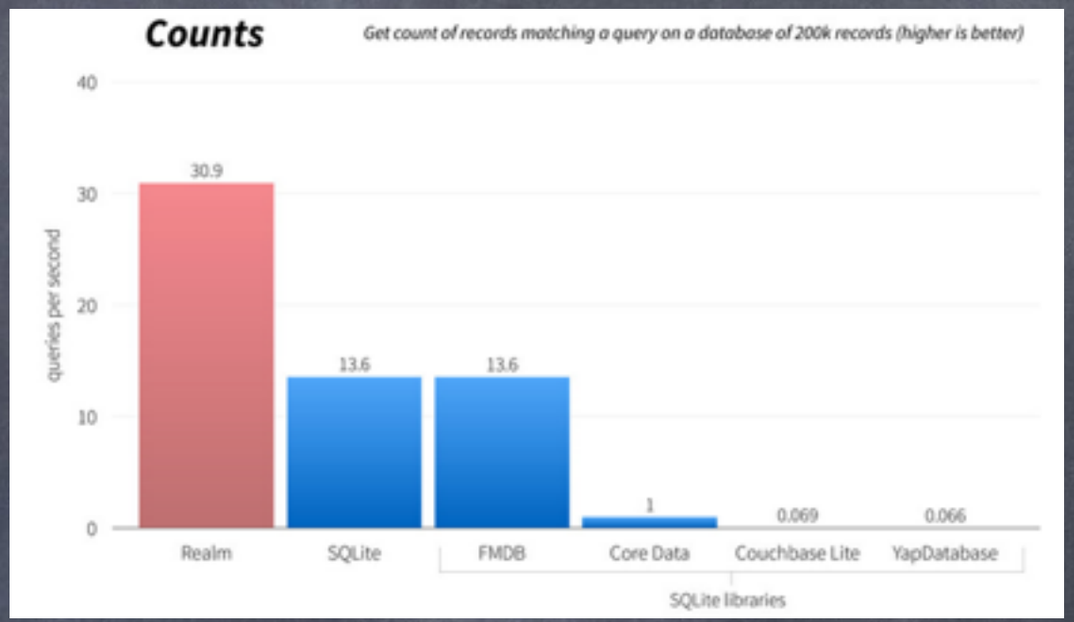
What is Realm?

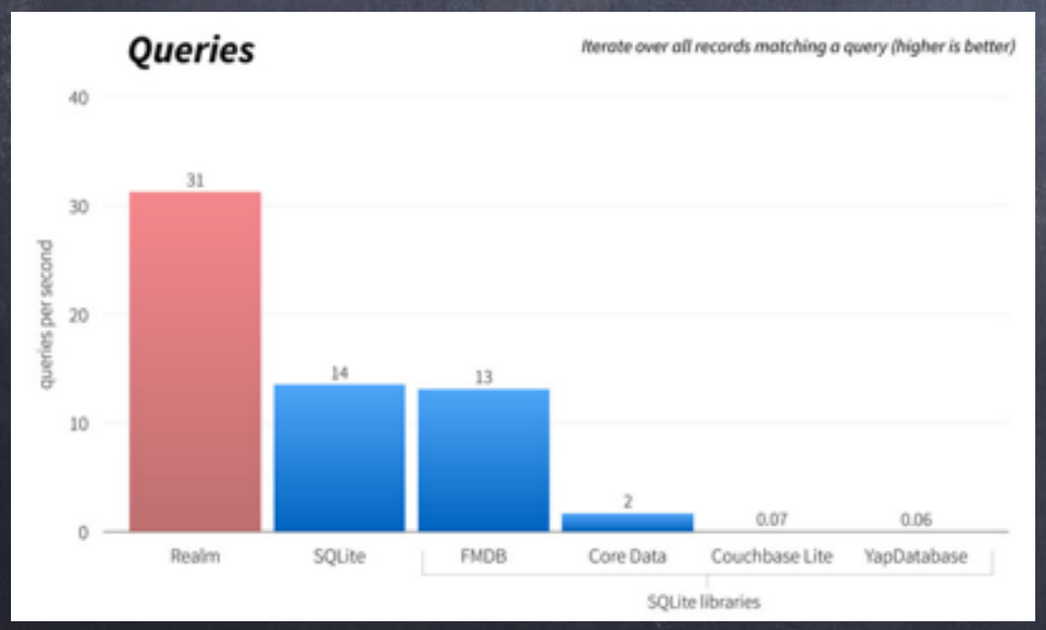
Realm is MAGIC!!!

- modern database written for mobile constraints
  - replacement for SQLite and/or Core Data
  - (also supports Android)
- Fast & Scalable
- Object-based
  - queries return objects & relationships to other objects
- Open Source
- CocoaPod

How fast?



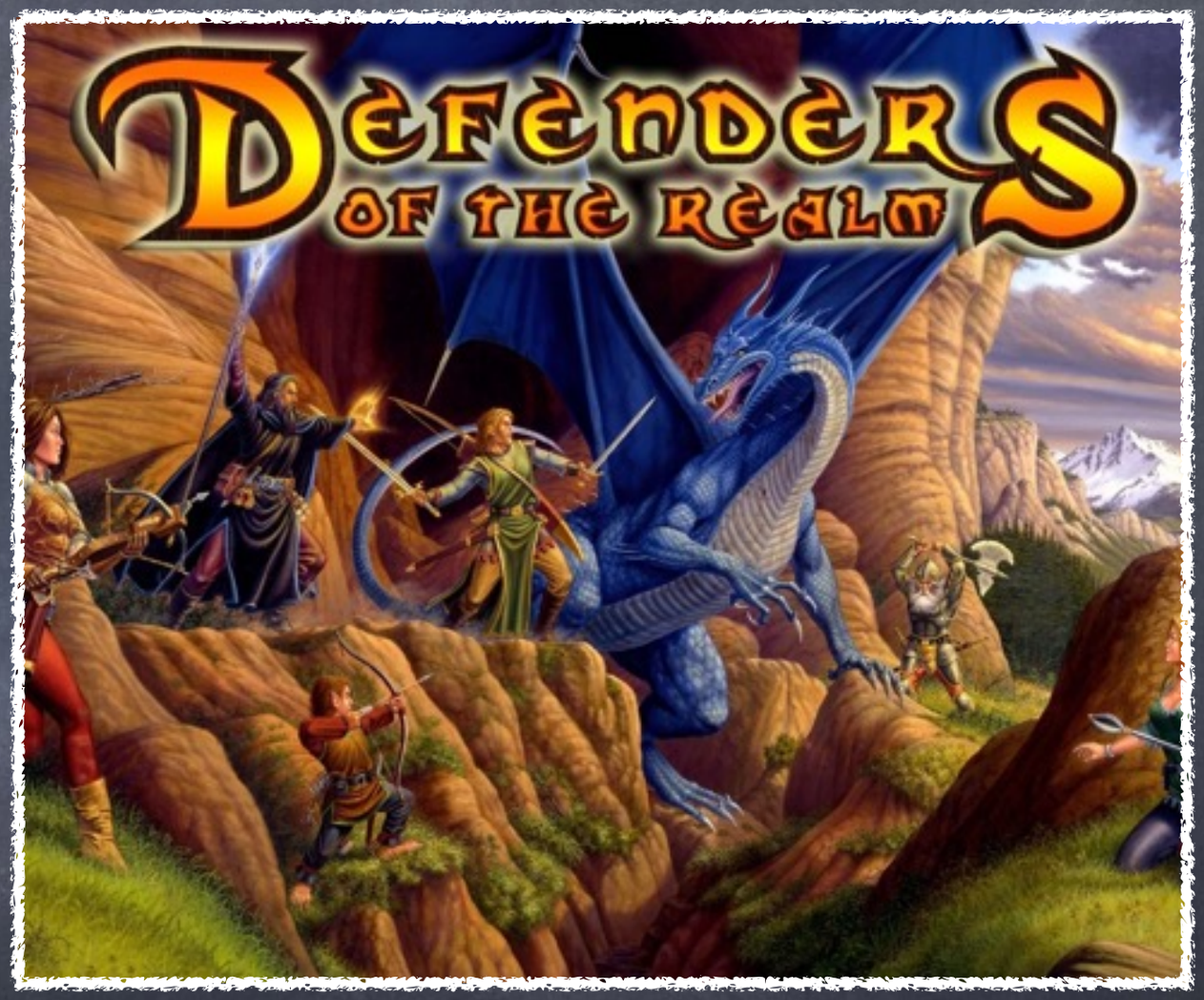




How? bit-packing, caching, vectorization and a zero-copy architecture

* Source: realm.io, though their benchmark code is available for download and scrutiny.

Why should you use it?



- Nice documentation*
- Almost no boilerplate code needed
- Incredibly easy to use
- Cross Platform
- Thread Safe
- Supports encryption**
- It's OMG FREE!!!

* Documentation is available in Objective-C or Swift

** Encryption adds ~10% overhead

General Mobile Database Use Cases

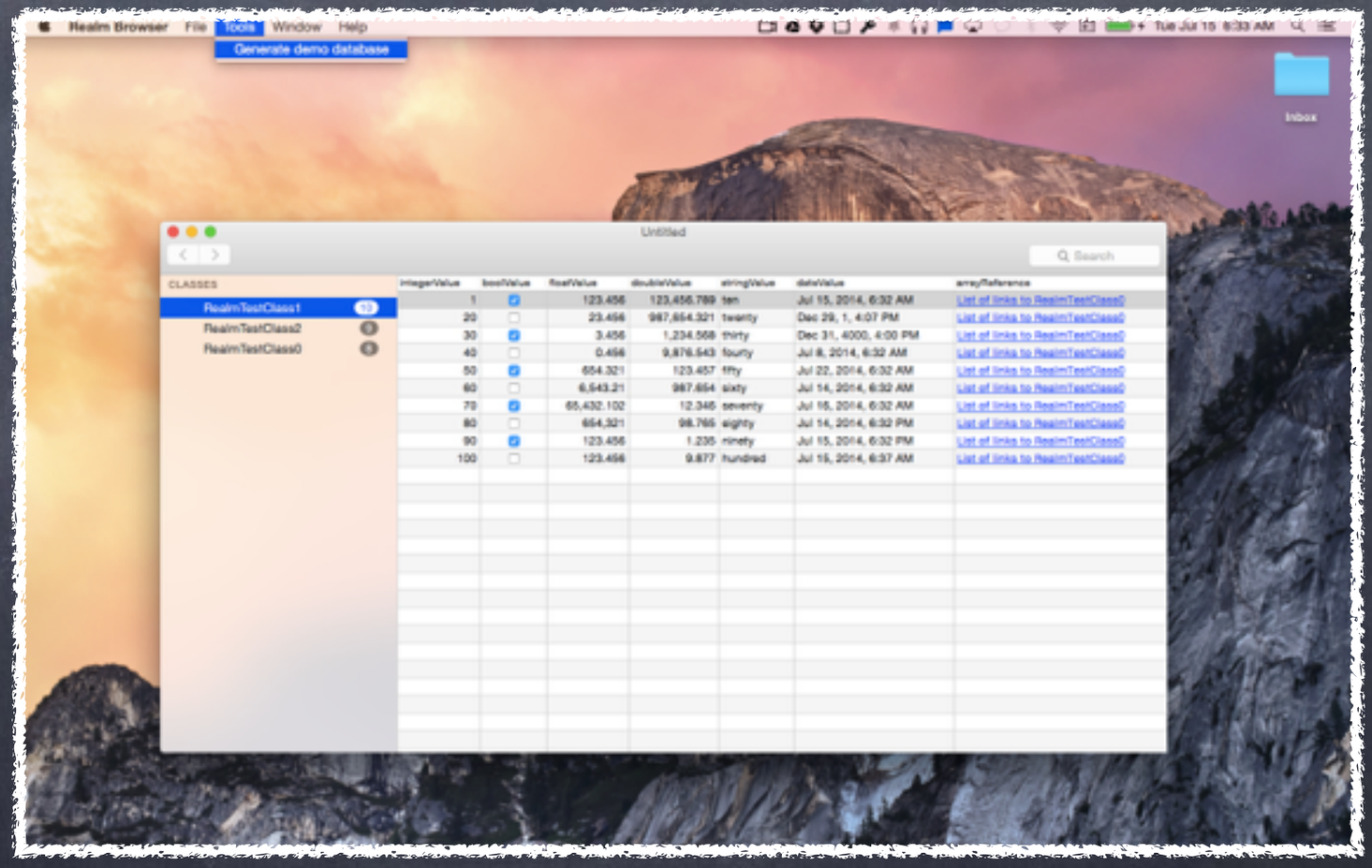
- Local Storage
  - Ex: grocery list, health tracker
- Cache for remote data
  - Ex: search history, RSS reader
- Pre-loaded Data
  - Ex: Recipe book, trivia game

Realm Use Case Advantages

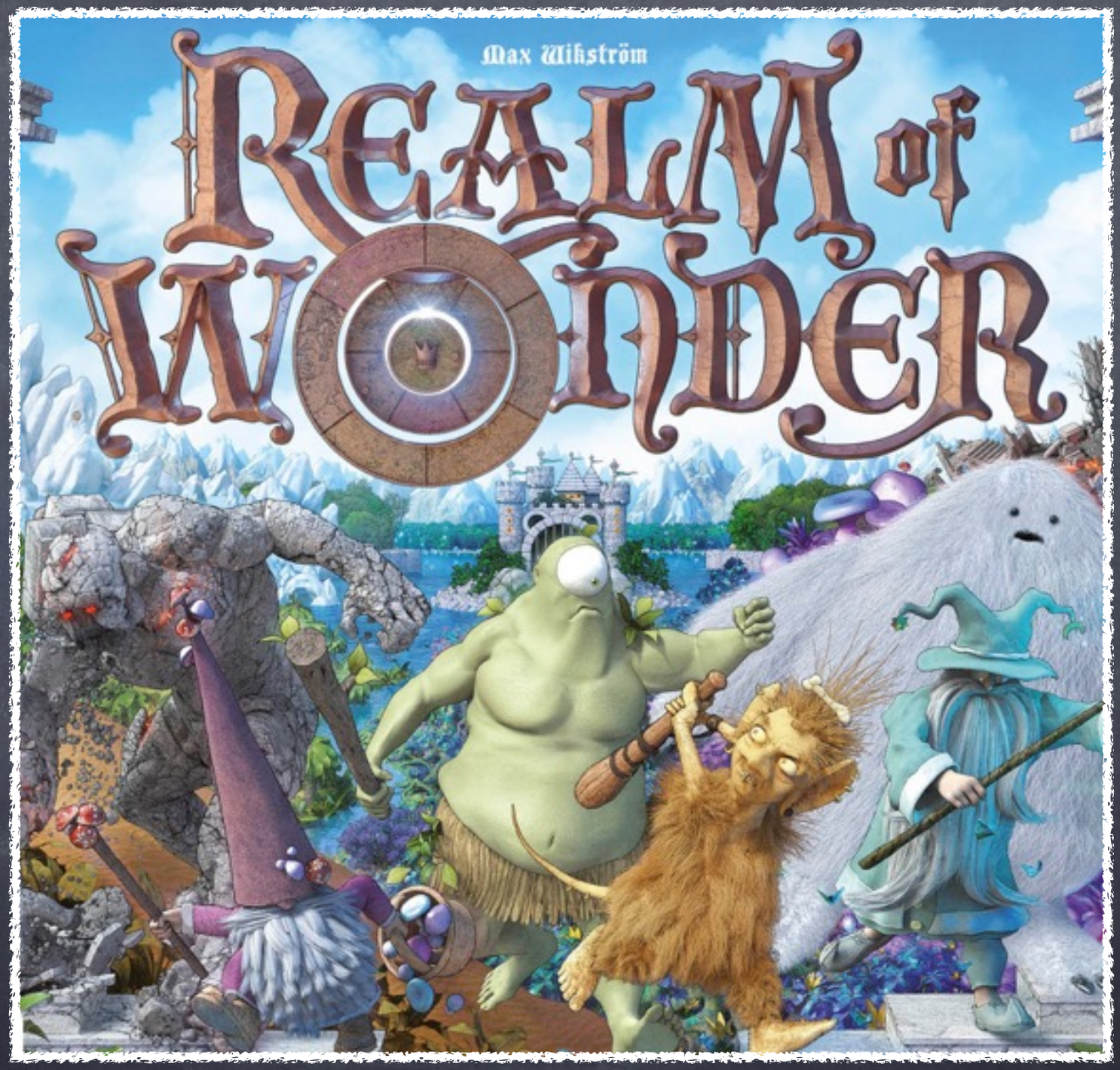
- Local Storage
  - define your objects, save them in 3 lines of code, from anywhere (thread independent)
- Cache for remote data
  - instantiate objects from NSDictionary, easy insert-or-update methods (using primary keys)
- Pre-loaded Data
  - Realm files are small, migrating schema (making changes to your db objects) is easy

Realm Browser

- desktop app
- functional



Code Examples



Tables are Objects & Rows are Instances

- Objects are subclasses of RLMObject
- Supported property types:
    - NSString
    - NSInteger, CGFloat, int, long, float, and double
    - BOOL or bool
    - NSDate
    - NSData
    - RLMObject subclasses, so you can have many-to-one relationships.
    - RLMArray<X>, where X is an RLMObject subclass, so you can have many-to-many relationships.

Class Examples

AB_Dog.h

```
#import <Realm/Realm.h>

@class AB_Person;

// Dog model
@interface AB_Dog : RLMObject
@property NSString    *name;
@property AB_Person   *owner;
@end

RLM_ARRAY_TYPE(AB_Dog)
// defines RLMArray<AB_Dog>
```

required for arrays

AB_Person.h

```
#import <Realm/Realm.h>

@class AB_Dog;

// Person model
@interface AB_Person : RLMObject
@property NSString      *name;
@property NSDate        *birthdate;
@property RLMArray<AB_Dog> *dogs;
@end
```

AB_Dog.m

```
#import “AB_Dog.h”

@implementation AB_Dog
@end // none needed
```

AB_Person.m

```
#import “AB_Person.h”

@implementation AB_Person
@end // none needed
```

Note that realm objects ignore property attributes – (nonatomic, strong, etc)

Optional Class Methods

```objc
@interface Book : RLMObject
@property float price;
@property NSString *title;
@end

@implementation Book
+ (NSDictionary *)defaultPropertyValues {
    return @{@"price" : @0, @"title": @""};
}
@end
```

default values

```objc
@interface Person : RLMObject
@property NSInteger id;
@property NSString *name;
@end

@implementation Person
+ (NSString *)primaryKey {
    return @"id";
}
@end
```

primary key

See also:
ignoredProperties, and
indexedProperties.

Inserting Objects

```objc
// Create object
Person *author = [[Person alloc] init];
author.name    = @"David Foster Wallace";

// Get the default Realm
RLMRealm *realm = [RLMRealm defaultRealm];
// You only need to do this once (per thread)

// Add to Realm with transaction
[realm beginWriteTransaction];
[realm addObject:author];
[realm commitWriteTransaction];
```

write transactions

```objc
// Create object
Person *author = [[Person alloc] init];
author.name    = @"David Foster Wallace";

// Get the default Realm
RLMRealm *realm = [RLMRealm defaultRealm];
// You only need to do this once (per thread)

// Add to Realm with asyncronous write transaction
[realm transactionWithBlock:^{
    [realm addObject:author];
}];
```

asynchronous, even

Updates

```objc
// Update an object with a transaction
[realm beginWriteTransaction];
author.name = @"Thomas Pynchon";
[realm commitWriteTransaction];
```

note that all changes MUST be in a write transaction

```objc
//Creating a book with the same primary key as a previously saved book
Book *cheeseBook = [[Book alloc] init];
cheeseBook.title = @"Cheese recipes";
cheeseBook.price = @9000;
cheeseBook.id = @1;

//Updating book with id = 1
[realm beginWriteTransaction];
[Book createOrUpdateInRealm:realm withObject:cheeseBook];
[realm commitWriteTransaction];
```

insert or update (based on primary key)

DESTRUCTION!

```objc
Book *cheeseBook = ... //Book we made previously

// Update an object with a transaction
[realm beginWriteTransaction];
[realm deleteObject:cheeseBook];
[realm commitWriteTransaction];
```

so easy!

Querying

```
// Query the default Realm
RLMResults *dogs = [Dog allObjects];
```

query using class methods

WHERE clause is essentially an NSPredicate string

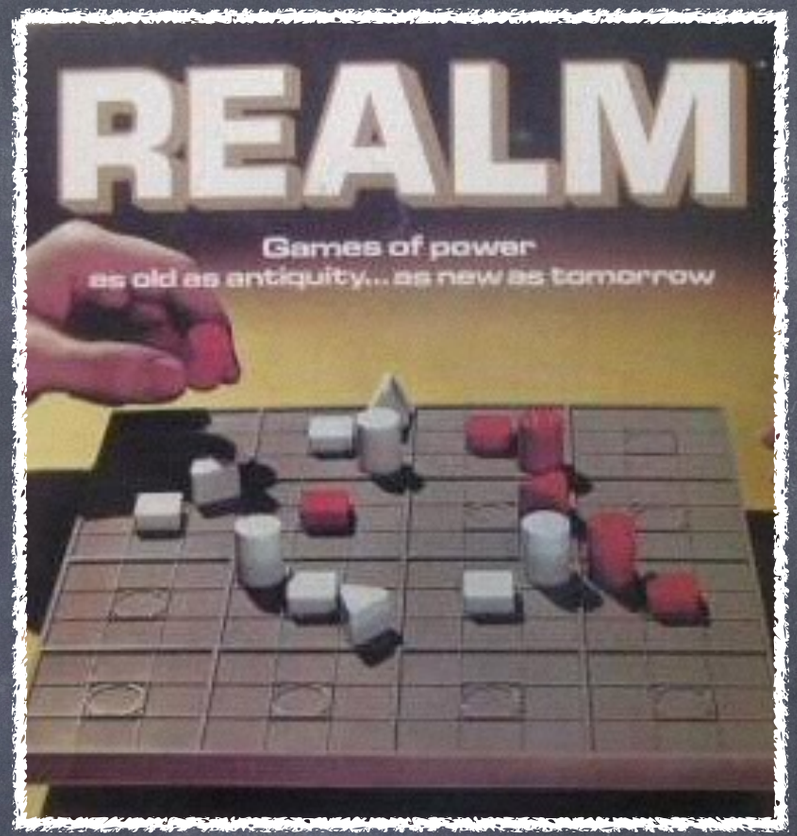
```
// Query using a predicate string
RLMResults *tanDogs = [Dog objectsWhere:@"color = 'tan' AND name BEGINSWITH 'B'"];

// Query using an NSPredicate object
NSPredicate *pred = [NSPredicate predicateWithFormat:@"color = %@ AND name BEGINSWIT
                                                     @"tan", @"B"];
tanDogs = [Dog objectsWithPredicate:pred];
```

```
// Sort tan dogs with names starting with "B" by name
RLMResults *sortedDogs = [[Dog objectsWhere:@"color = 'tan' AND name BEGINSWITH 'B'"]
                                    sortedResultsUsingProperty:@"name" ascending:YES];
```

sorting is also easy!

Memory Matters Trivia App DEMO



- pre-loaded data
- data conversion from .plist
- encapsulation of Realm access
- editing UI

Sources

- Realm.io
  - Nice introduction: http://realm.io/news/introducing-realm/
  - Obj-C/Swift Documentation: http://realm.io/docs/cocoa/
  - Obj-C API: http://realm.io/docs/cocoa/0.91.1/api/
  - github: https://github.com/realm/realm-cocoa
- NSHipster - http://nshipster.com/nspredicate/
- Images*
  - Magic Realm: https://www.boardgamegeek.com/boardgame/22/magic-realm
  - Defenders of the Realm: https://www.boardgamegeek.com/boardgame/65532/defenders-realm
  - Realm of Wonder: https://www.boardgamegeek.com/boardgame/162580/realm-wonder
  - Realm: https://www.boardgamegeek.com/boardgame/3024/realm
  - Realm of Heroes: https://www.boardgamegeek.com/boardgame/139401/realm-heroes

* (I make no promises that any of these games are any good.)

Thank you.

Martin Grider

@livingtech

blog: http://chesstris.com/
business: http://abstractpuzzle.com/