

# Core Data

**Managed Object Model:** Define the data you want to store.

- Entity: This is an individual data object. In a database it'd be a **table**.
- Attribute. This is an individual variable stored inside an entity. In a database it'd be a **column**.

**Managed Object:** The object representation of an entity you defined. You create, modify, and store those while using core data.

**Managed Object Context:** Sort of a temporary storage area. You take entities from the persistent store and put them here to modify them. After you're done with them, they will be safely put back in the persistent store.

**Persistent Storage Coordinator:** Coordinates your managed object model and the persistent store. You just set it up and it takes care of everything.

- **Insert data:**

```
MyData *myData = (MyData *)[NSEntityDescription
    insertNewObjectForEntityForName:@"MyData"
    inManagedObjectContext:objCon];
```

- Insert a new object for a specified entity. This corresponds to inserting a row into a table.
- You need to know the class representing your entity (MyData) and the name of the entity in the persistent store ("MyData").
- After myData was created, assign its properties to the values you want it to store. (Like myData.name = @"foo".)
- objCon is your managed object context. Call its save method when you want the new object to be actually inserted.

- **Retrieve data (ALL objects of an entity):**

```
NSFetchRequest *fetchRequest = [[NSFetchRequest alloc] init];
NSEntityDescription *entity = [NSEntityDescription entityForName:@"MyData"
    inManagedObjectContext:objCon];
[fetchRequest setEntity:entity];
NSError *error;
NSArray *items = [objCon executeFetchRequest:fetchRequest error:&error];
```

- Create a new NSFetchRequest.
- Create a NSEntityDescription. This means to tell the managed object context what entity you want to retrieve.
- The fetchRequest also needs to know this entity.
- Start the retrieval of all objects of the entity by executing the fetchRequest. This returns an array of MyData objects you can now work with. Keep in mind that it may be empty.

- **Retrieve data (only selected objects of an Entity):**

```
NSPredicate *predicate = [NSPredicate predicateWithFormat:@"id=%i", id];
[fetchRequest setPredicate:predicate];
```

- After you created a fetchRequest, create a NSPredicate.
- The predicate consists of a formatted string which is like the where-clause of SQL.
- Tell the fetchRequest to use that predicate to filter the objects it retrieves when executed.

- **Delete data:**

```
[objCon deleteObject:managedObject];
```

- Use a fetchRequest to retrieve an object like myData which you want to delete.
- Call the deleteObject method on the managed object context, tell it which managed object you want to be deleted.
- Don't forget to save the objCon. This actually deletes the myData object.

- **Sort data on retrieval:**

```
NSSortDescriptor *sort = [[NSSortDescriptor alloc] initWithKey:@"myAttribute"
    ascending:YES];
[fetchRequest setSortDescriptors:[NSArray arrayWithObject:sort]]; [sort release];
```

- Create a fetchRequest, an entity description and a predicate.
- Create a NSSortDescriptor, which you tell the attribute used for sorting, and the order (ascending:NO for descending order).
- Tell the fetchRequest to use that sort descriptor. It is capable of using more than one, so you have to put it in an array.
- Execute the fetchRequest as usual. The resulting array is now sorted.