

## Base de Datos (Sqlite)

### Implementando Sqlite

- **Introducción a sqlite**
- **Implementar libsqlite3.dylib framework**
  - verificar si existe la bd en el filesystem
  - copiar desde recursos a filesystem , implementar metodos
- **Realizar consulta**
  - abrir bd- conexion
  - Consulta sql
  - cerrar bd
- **Tipos de consultas comunes**
  - Select
  - Insert
  - Delete
  - Update



Una **base de datos** o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso

### SQLite 3

**SQLite es un sistema de gestión de bases de datos relacional compatible con ACID** (Cuando se dice que es ACID compliant se indica -en diversos grados- que éste permite realizar transacciones.)

Contiene un propio motor y no requiere un servidor para funcionar.  
Se puede usar en cualquier plataforma y es portable lo cual permite usar en móviles

-Funciona con instrucciones SQL

-Esta soportado por la comunidad [www.sqlite.org](http://www.sqlite.org)

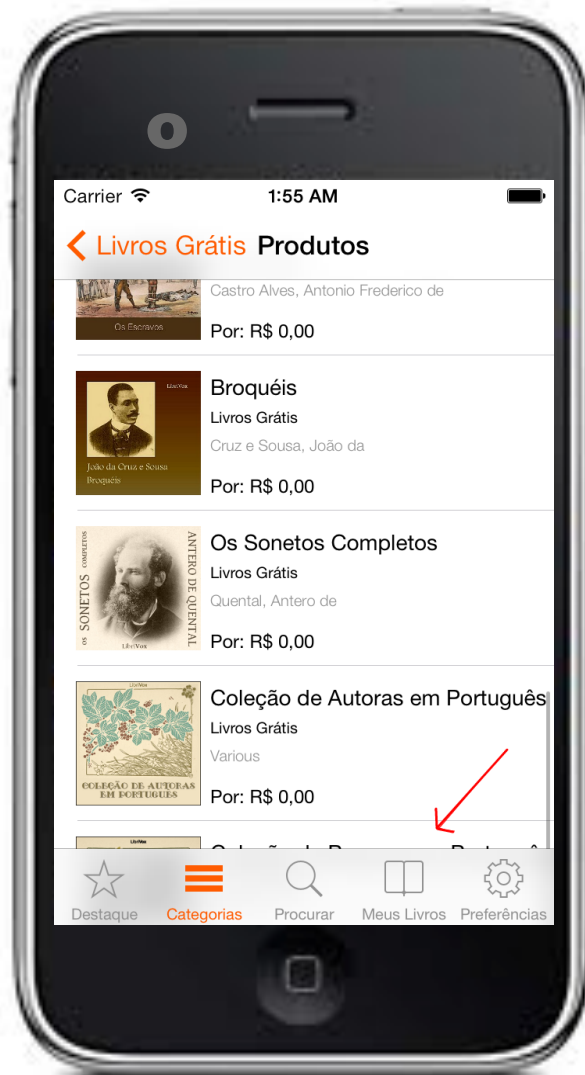
#### Que es SQL:

Es una **lenguaje estructurado** usado para definir e interactuar con datos los cuales se almacenan en tablas

**Tablas:** contienen columnas y registros

**Columnas:** nombre de campo, define el tipo de datos

**Registro:** es un set de columnas el cual permite crear una fila de datos en la tabla



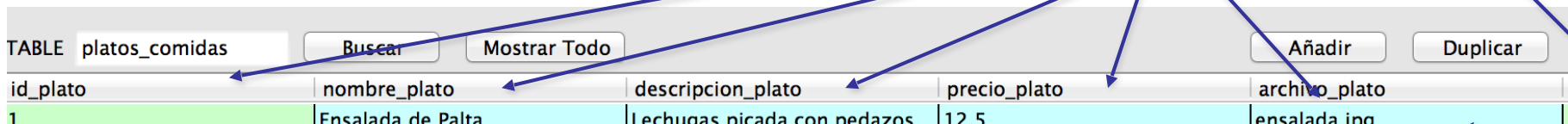
BD -sqlite



## Sqlite Manager

Base de datos: es un conjunto de tablas

Tabla: colección de datos ordenados por campos y registros



id_plato	nombre_plato	descripcion_plato	precio_plato	archivo_plato
1	Ensalada de Palta	Lechugas picada con pedazos...	12.5	ensalada.jpg
2	Ensalada de Mixta	Lechugas picada con pedazos...	12.5	ensalada2.jpg
3	Ceviche de Mero	Pescado en trozos marinado ...	30.3	ceviche.jpg
4	Causa de pollo	papa molida con aji con relle...	12.5	causa.jpg
5	Anticucho de corazón	trozos de carne de corazon a ...	14.5	anticucho.jpg
6	Anticucho de corazón	trozos de carne de corazon a ...	22.5	anticucho.jpg
7	Lomo Saltado de Carne	trozos de carne con papas fri...	32.5	lomo_saltado.jpg
8	Lomo Saltado de Pollo	trozos de carne de pollo con ...	32.5	lomo_saltado.jpg

A- crear la base de datos

B- crear una tabla

C- llenar la tabla con datos



# Sqlite Manager, DbBrowser

Descargar desde : <http://sqlitebrowser.org/>

## DB Browser for SQLite

The Official home of the DB Browser for SQLite

[View project on GitHub](#)

### // News

2017-09-28 - Added PortableApp version of 3.10.1. Thanks John. :)  
2017-09-20 - DB Browser for SQLite 3.10.1 has been released! :D  
2017-09-08 - Removed the continuous AppImage builds for Linux due to problems with the upload script.

### // Screenshot



	list	month	members
1	gluster-board	2013-09-05	99999
2	gluster-users	2013-09-05	99999

Download 32-bit Windows .exe

Download 64-bit Windows .exe

Download PortableApp

Download Mac .dmg

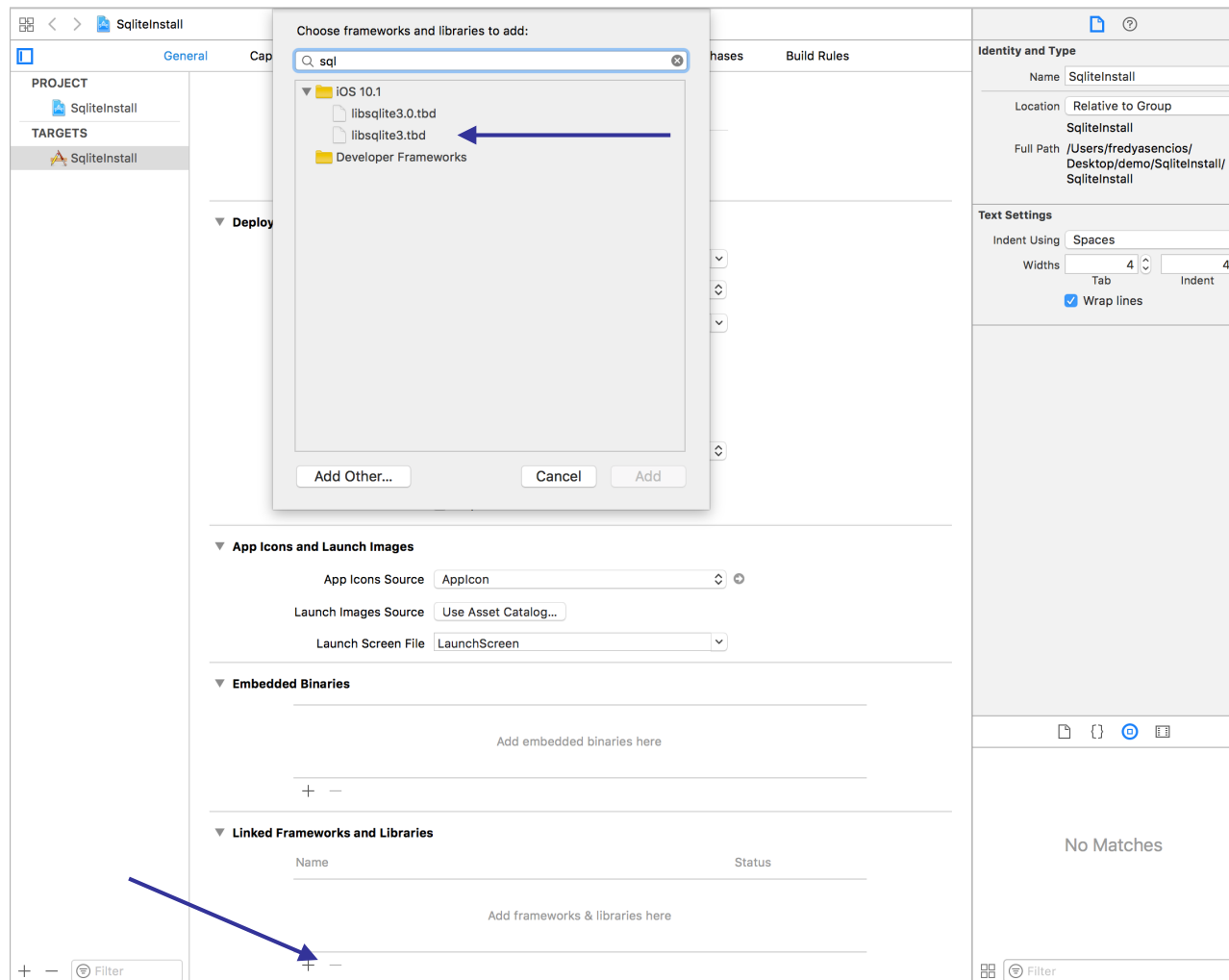
Download source .zip

Download tar.gz file



# Sqlite Implementación básica

## 1- importar SQLite libreria desde frameworks





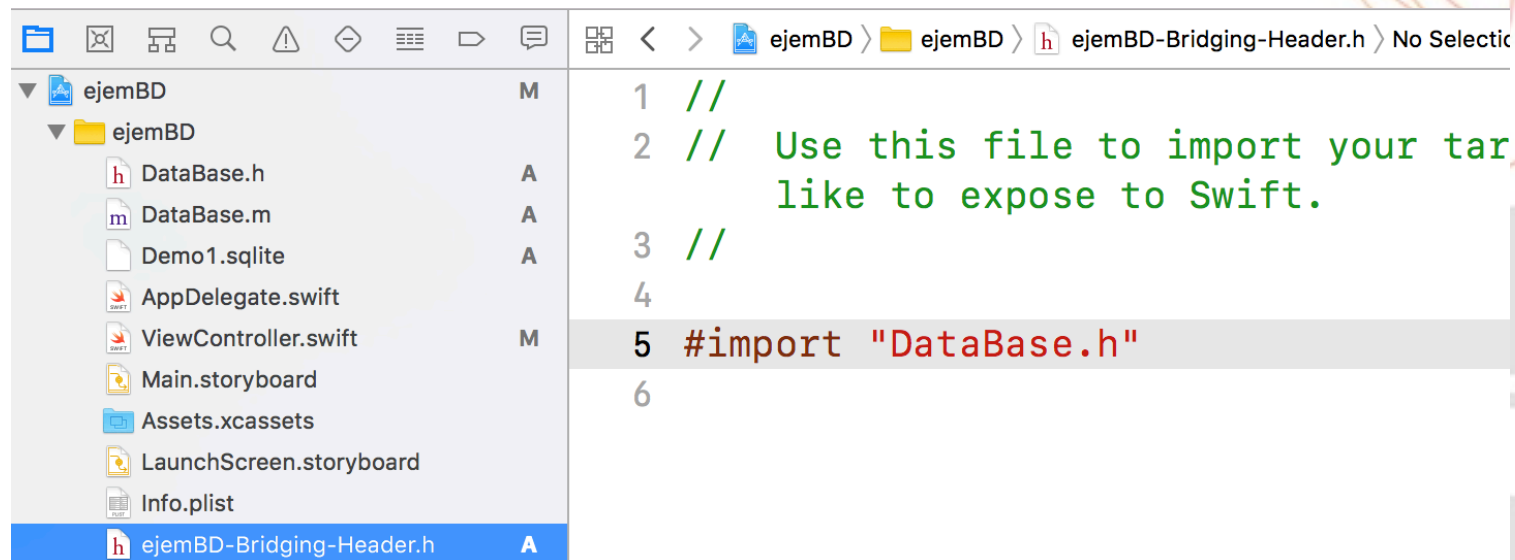
## Sqlite Implementación básica en iOS

2- arrastrar los ficheros de libreria

DataBase.h

DataBase.m

3- aceptar la creación del fichero bridge



## Sqlite Implementación básica en iOS

4- importar en el fichero header (h) DataBase.h

Usando el tag

```
#import "DataBase.h"
```

5- usar el método

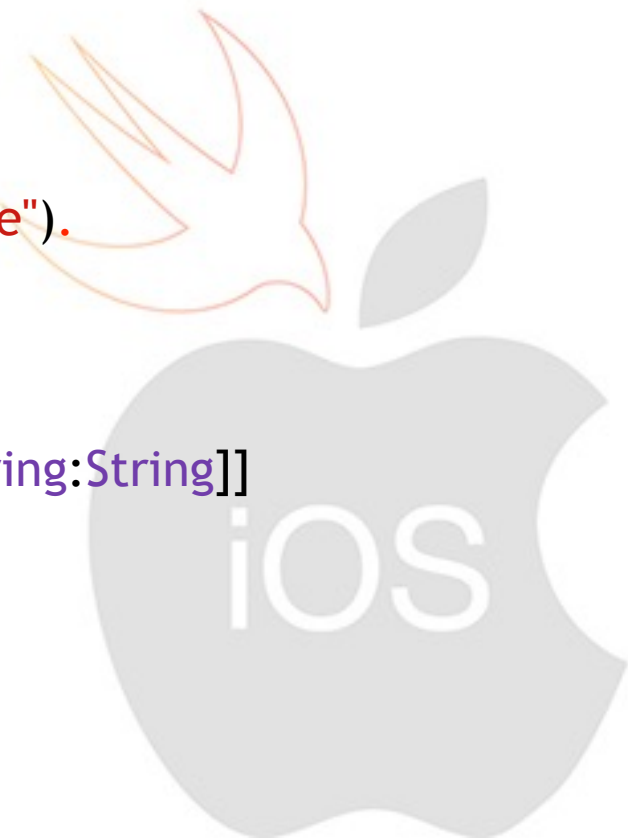
checkAndCreate para la instalación en el proyecto

```
DataBase.checkAndCreateDatabase(withName: "Demo1.sqlite").
```

6- usar el método de lectura de tabla (select)

Ejem:

```
DataBase().ejecutarSelect("Select * from alumnos ") as! [[String:String]]
```



## MODELO CONSULTA SQL “Select ...”

Select \* FROM nombre\_tabla ...

```
-(void)leerTablaBD
{
    [self openDatabase];
    NSString *sql=@"select * from platos_comidas order by id_categoria ";
    NSLog(@"sql: %@",sql);

    sqlite3_stmt *compiledStatement;
    if(sqlite3_prepare_v2(base,[sql UTF8String], -1, &compiledStatement, nil)== SQLITE_OK){
        while (sqlite3_step(compiledStatement) == SQLITE_ROW){

            NSString *idPlato = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 0)] ;
            NSString *nombrePlato = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 1)];
            NSString *descripPlato = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 2)];
            NSString *precioPlato = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 3)];
            NSString *archivoPlato = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 4)];
            NSString *idCategoria = [NSString stringWithUTF8String:(char *)sqlite3_column_text(compiledStatement, 5)];

        }
    }
}
```





## MODELO CONSULTA SQL UPDATE...

UPDATE nombre\_tabla SET nom\_campo = 'valor '

```
- (void)actualizarTablaxID:(NSString *)IDPL newNombre:(NSString *)NOMNEW  
{
```

```
    [self openDatabase];  
    NSString *sql =[NSString stringWithFormat:@"UPDATE platos_comidas SET  
nombre_plato ='%@' WHERE id_plato='%@'",NOMNEW,IDPL];
```

```
    char *err;  
    if(sqlite3_exec(base,[sql UTF8String], NULL, NULL, &err)!= SQLITE_OK)  
    {  
        sqlite3_close(base);  
    }  
}
```



## MODELO DE CONSULTA SQL BORRAR todos los registros

**“DELETE FROM nombre\_tabla”**

```
- (void)BorrarContenidoTabla  
{
```

```
    [self openDatabase];  
    NSString *sql=@"DELETE FROM platos_comidas";
```

```
    NSLog(@"sql: %@",sql);
```

```
    char *err;
```

```
    if(sqlite3_exec(base,[sql UTF8String], NULL, NULL, &err)!= SQLITE_OK){
```

```
        sqlite3_close(base);
```

```
        NSLog(@"Error al Borrar");
```

```
    }
```

```
}
```



## MODELO DE CONSULTA SQL BORRAR con parametro de entrada

**“DELETE FROM nombre\_tabla WHERE nom\_campo = ‘valor ’”**

```
- (void)borrarContenidoTablaxID:(NSString *)IDPL
{

    [self openDatabase];
    NSString *sql =[NSString stringWithFormat:@"DELETE FROM platos_comidas
    WHERE id_plato='%@'",IDPL];

    NSLog(@"sql: %@",sql);
    char *err;
    if(sqlite3_exec(base,[sql UTF8String], NULL, NULL, &err)!= SQLITE_OK){
        sqlite3_close(base);
        NSLog(@"Error al Borrar");
    }
}
```



## MODELO DE CONSULTA SQL INSERT con parametros de entrada

“INSERT INTO nombre\_tabla WHERE (nom\_campo,... ) VALUES (valor,...)”

```
- (void)grabarDatosComida:(NSString *)NOMB descripcion:(NSString *)DESC
precio:(NSString *)PREC archivo:(NSString *)ARCH categoria:(NSString *)CATE
{
    [self openDatabase];

    NSString *sql =[NSString stringWithFormat:@"insert into platos_comidas
(nombre_plato', 'descripcion_plato', 'precio_plato',
'archivo_plato','id_categoria') values
('%@','%@','%@','%@','%@')",NOMB,DESC,PREC,ARCH,CATE];
    NSLog(@"sql: %@",sql);
    char *err;
    if(sqlite3_exec(base,[sql UTF8String], NULL, NULL, &err)!= SQLITE_OK){
        sqlite3_close(base);
        NSLog(@"Error al insertar");
    }
}
```



Ejemplo práctico con el objetivo siguiente

- implementando Sqlite con Capa MODEL

Tema a usar : menú comidas , restaurant

