作者 JohnnyB0Y (/users/8939e3430d49) 2015.08.29 16:12*
写了18440字，被83人关注，获得了55个喜欢
(/users/8939e3430d49)

iOS开发--SQLite数据库

字数1088 阅读2679 评论4 喜欢3

这是我个人的学习笔记，如有不同见解欢迎评论交流 .
(我的微博 : http://weibo.com/JohnnyB0Y (http://weibo.com/JohnnyB0Y))

> ## 简单介绍
>
> - **SQLite**
>
> 1.SQLite是一个由C语言编写的自包含的SQL关系型数据库引擎 .
>
> 2.如果只需要关系型数据库提供的功能请直接使用SQLite .
> - 更轻量级的FMDBatabase对SQLite进行了封装，使用方便简单。

-

SQLite的命令行使用

- **创建数据库**

  1.打开命令行后输入 sqlite3 catalog.db , 启动命令行工具并创建数据库 .

  2.使用 ATTACH DATABASE 命令把多个数据库添加到命令行工具 , 从而操纵数据 .

  3.创建表

```
CREATE TABLE "main"."Product" ("ID" INTEGER PRIMARY KEY AUTOINCREMENT NOT NU
"Name" TEXT, "ManufacturerID" INTEGER, "Details" TEXT, "Price" DOUBLE, "Quan
INTEGER, "CountryOfOriginID" INTEGER, "Image" TEXT);
```

  4.填充数据库

```
INSERT INTO "main"."Product" ("Name","ManufacturerID","Details","Price","Qua
"CountryOfOriginID","Image") VALUE ('Widget A','1','Details of Widget','1.29
```

  4.1把编辑好的文件批量导入数据库

```
输入命令 .separator "\t" 指定 \t 作为数据文件中字段的分隔符 ,
然后输入 .import "Products.txt" Product , 导入Products.txt文件到Product表中 .
```

  5.读取行数据

```
SELECT * FROM country; 查找country表中所有数据。
SELECT NAME,PRICE FROM product ORDER BY price; 查看使用价格排序的所有产品。
SELECT NAME,COUNTRY FROM Product, country where product.countryoforiginid=co
```

-

SQLite的使用

- 打开现有数据库

```
/** 要操作的数据库文件路径 */
+ (NSString *) pathWithDatabase
{
    NSArray *paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,
                                                         NSUserDomainMask,
                                                         YES);
    NSString *documentDirectory = [paths lastObject];
```

```
    return [documentDirectory stringByAppendingPathComponent:@"catalog.db"];
}
/** 创建数据库 */
- (void) createEditableDatabase
{
    BOOL success;
    NSError *error;
    NSFileManager *fileManager = [NSFileManager defaultManager];
    // 数据库文件路径
    NSString *writableDB = [DBAccess pathWithDatabase];
    // 文件是否存在
    success = [fileManager fileExistsAtPath:writableDB];
    if (!success) {
        // 不存在就创建
        NSString *defaultPath = [[[NSBundle mainBundle]
                                          resourcePath]
          stringByAppendingPathComponent:@"catalog.db"];
        // 拷贝文件到某文件路径
        success = [fileManager copyItemAtPath:defaultPath
                                       toPath:writableDB
                                        error:&error];
        if (!success) {
            NSAssert1(0, @"Failed to create writable database file:'%@'.",
                      [error localizedDescription]);
        }
    }
}
/** 打开数据库 */
- (void)initializeDatabase
{
    // 确认可操作数据是否存在
    [self createEditableDatabase];
    // 数据库路径
    NSString *path = [DBAccess pathWithDatabase];
    // 是否打开成功
    if (sqlite3_open([path UTF8String], &database) == SQLITE_OK)
    {
        NSLog(@"Opening Database");
    }
    else
    {
        // 打开数据库失败
        sqlite3_close(database);
        NSAssert1(0, @"Failed to open database: '%s'.", sqlite3_errmsg(databa
    }
}
```

- 查询数据库

```
- (NSMutableArray *)getAllProducts
{
    // 查询语句
    char *const sql = "SELECT product.ID, product.Name, Manufacturer.name, \
    product.details, product.price, product.quantityonhand, country.country, \
    product.image FROM Product, Manufacturer, Country WHERE \
    manufacturer.manufacturerid=product.manufacturerid AND \
    product.countryoforiginid=country.countryid";
    // 将sql文本转换成一个准备语句
    sqlite3_stmt *statement;
    int sqlResult = sqlite3_prepare_v2(database, sql, -1, &statement, NULL);
    // 装查询结果的可变数组
    NSMutableArray *arrayM = [NSMutableArray array];
    // 结果状态为OK时，开始取出每条数据
    if ( sqlResult == SQLITE_OK) {
        // 只要还有下一行，就取出数据。
        while (sqlite3_step(statement) == SQLITE_ROW) {
            Product *product = [[Product alloc] init];
            // 每列数据
            char *name = (char *)sqlite3_column_text(statement, 1);
            char *manufacturer = (char *)sqlite3_column_text(statement, 2);
            char *details = (char *)sqlite3_column_text(statement, 3);
            char *countryOfOrigin = (char *)sqlite3_column_text(statement, 6);
            char *image = (char *)sqlite3_column_text(statement, 7);
            // 为模型赋值
            product.ID = sqlite3_column_int(statement, 0);
            product.name = [self stringWithCharString:name];
            product.manufacturer = [self stringWithCharString:manufacturer];
            product.details = [self stringWithCharString:details];
            product.price = sqlite3_column_double(statement, 4);
            product.quantity = sqlite3_column_int(statement, 5);
            product.countryOfOrigin = [self stringWithCharString:countryOfOrig
            product.image = [self stringWithCharString:image];
            // 添加进数组
            [arrayM addObject:product];
        }
        // 完成后释放prepare创建的准备语句
        sqlite3_finalize(statement);
    }
    else
    {
        NSLog(@"Problem with database:");
        NSLog(@"%d",sqlResult);
    }
    return arrayM;
}
```

```
/** C字符串转换OC字符串 */
- (NSString *) stringWithCharString:(char *)string
{
    return (string) ? [NSString stringWithUTF8String:string] : @"";
}
```

- 关闭数据库

```
- (void)closeDatabase
{
    if (sqlite3_close(database) != SQLITE_OK) {
        NSAssert1(0, @"Failed to close database: '%s'.", sqlite3_errmsg(data
    }
}
```

- 参数化查询

```
{
      // 条件查询举例
      NSString *sql = @"SELECT Product.name, country.country
                        FROM country, product
                        WHERE country=? and countryoforiginid=? ";
      // 查询语句绑定函数，可以运行时动态设置成需要查询的值（可缓存和重用，性能好）。
      // 第一个参数为预编译语句，第二个为SQL语句中‘? ’的索引（从1开始），第三个为运行时确
      sqlite3_bind_blob(<#sqlite3_stmt *#>, <#int#>, <#const void *#>, <#int
      sqlite3_bind_double(<#sqlite3_stmt *#>, <#int#>, <#double#>);
      sqlite3_bind_int(<#sqlite3_stmt *#>, <#int#>, <#int#>);
      sqlite3_bind_int64(<#sqlite3_stmt *#>, <#int#>, <#sqlite3_int64#>);
      sqlite3_bind_null(<#sqlite3_stmt *#>, <#int#>);
      sqlite3_bind_text(<#sqlite3_stmt *#>, <#int#>, <#const char *#>, <#int
      sqlite3_bind_text16(<#sqlite3_stmt *#>, <#int#>, <#const void *#>, <#i
      sqlite3_bind_value(<#sqlite3_stmt *#>, <#int#>, <#const sqlite3_value
      sqlite3_bind_zeroblob(<#sqlite3_stmt *#>, <#int#>, <#int n#>);
}
```

-

FMDBatabase的使用

占坑

推荐拓展阅读 (/sign_in)

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作！

¥ 打赏支持

♡ 喜欢 | 3

分享到微博 分享到微信
更多分享

4条评论 （ 按时间正序 · 按时间倒序 · 按喜欢排序 ）

添加新评论 (/sign_in)

土豆和香菇 (/users/4fc4a90a0c0f)
2楼 · 2016.01.06 15:22 (/p/d7ec0acf646a/comments/1172406#comment-1172406)
(/users/4fc4a90a0c0f)
这本书好眼熟!,,,,,,,还是蛮不错的

♡ 喜欢(0) 回复

JohnnyB0Y (/users/8939e3430d49)： @FLC_FY (/users/4fc4a90a0c0f) 是啊，老外写的都非常不错。
2016.01.06 17:26 (/p/d7ec0acf646a/comments/1173186#comment-1173186) 回复

阿磁 (/users/b2a57f42633f)： @FLC_FY (/users/4fc4a90a0c0f) 那本书? 😦
2016.08.24 09:07 (/p/d7ec0acf646a/comments/3803978#comment-3803978) 回复

添加新回复

土豆和香菇 (/users/4fc4a90a0c0f)
3楼 · 2016.01.07 20:23 (/p/d7ec0acf646a/comments/1181294#comment-1181294)
(/users/4fc4a90a0c0f)
我最近在看!

♡ 喜欢(0) 回复

登录后发表评论 (/sign_in)

(/sign_in)