SQLite Distinct 关键字 SQLite 约束

# SQLite PRAGMA

SQLite 的 **PRAGMA** 命令是一个特殊的命令，可以用在 SQLite 环境内控制各种环境变量和状态标志。一个 PRAGMA 值可以被读取，也可以根据需求进行设置。

## 语法

要查询当前的 PRAGMA 值，只需要提供该 pragma 的名字：

```
PRAGMA pragma_name;
```

要为 PRAGMA 设置一个新的值，语法如下：

```
PRAGMA pragma_name = value;
```

设置模式，可以是名称或等值的整数，但返回的值将始终是一个整数。

## auto_vacuum Pragma

**auto_vacuum** Pragma 获取或设置 auto-vacuum 模式。语法如下：

```
PRAGMA [database.]auto_vacuum;
PRAGMA [database.]auto_vacuum = mode;
```

其中，**mode** 可以是以下任何一种：

| Pragma 值 | 描述 |
| --- | --- |
| 0 或 NONE | 禁用 Auto-vacuum。这是默认模式，意味着数据库文件尺寸大小不会缩小，除非手动使用 VACUUM 命令。 |
| 1 或 FULL | 启用 Auto-vacuum，是全自动的。在该模式下，允许数据库文件随着数据从数据库移除而缩小。 |
| 2 或 INCREMENTAL | 启用 Auto-vacuum，但是必须手动激活。在该模式下，引用数据被维持，免费页面只放在免费列表中。这些页面可在任何时候使用 **incremental_vacuum pragma** 进行覆盖。 |

## cache_size Pragma

**cache_size** Pragma 可获取或暂时设置在内存中页面缓存的最大尺寸。语法如下：

```
PRAGMA [database.]cache_size;
PRAGMA [database.]cache_size = pages;
```

**pages** 值表示在缓存中的页面数。内置页面缓存的默认大小为 2,000 页，最小尺寸为 10 页。

## case_sensitive_like Pragma

**case_sensitive_like** Pragma 控制内置的 LIKE 表达式的大小写敏感度。默认情况下，该 Pragma 为 false，这意味着，内置的 LIKE 操作符忽略字母的大小写。语法如下：

```
PRAGMA case_sensitive_like = [true|false];
```

目前没有办法查询该 Pragma 的当前状态。

## count_changes Pragma

**count_changes** Pragma 获取或设置数据操作语句的返回值，如 INSERT、UPDATE 和 DELETE。语法如下：

```
PRAGMA count_changes;
PRAGMA count_changes = [true|false];
```

默认情况下，该 Pragma 为 false，这些语句不返回任何东西。如果设置为 true，每个所提到的语句将返回一个单行单列的表，由一个单一的整数值组成，该整数表示操作影响的行。

## database_list Pragma

**database_list** Pragma 将用于列出了所有的数据库连接。语法如下：

```
PRAGMA database_list;
```

该 Pragma 将返回一个单行三列的表格，每当打开或附加数据库时，会给出数据库中的序列号，它的名称和相关的文件。

## encoding Pragma

**encoding** Pragma 控制字符串如何编码及存储在数据库文件中。语法如下：

```
PRAGMA encoding;
PRAGMA encoding = format;
```

格式值可以是 UTF-8、UTF-16le 或 UTF-16be 之一。

## freelist_count Pragma

**freelist_count** Pragma 返回一个整数，表示当前被标记为免费和可用的数据库页数。语法如下：

```
PRAGMA [database.]freelist_count;
```

格式值可以是 UTF-8、UTF-16le 或 UTF-16be 之一。

# index_info Pragma

**index_info** Pragma 返回关于数据库索引的信息。语法如下：

```
PRAGMA [database.]index_info( index_name );
```

结果集将为每个包含在给出列序列的索引、表格内的列索引、列名称的列显示一行。

# index_list Pragma

**index_list** Pragma 列出所有与表相关联的索引。语法如下：

```
PRAGMA [database.]index_list( table_name );
```

结果集将为每个给出列序列的索引、索引名称、表示索引是否唯一的标识显示一行。

# journal_mode Pragma

**journal_mode** Pragma 获取或设置控制日志文件如何存储和处理的日志模式。语法如下：:

```
PRAGMA journal_mode;
PRAGMA journal_mode = mode;
PRAGMA database.journal_mode;
PRAGMA database.journal_mode = mode;
```

这里支持五种日志模式：

| Pragma 值 | 描述 |
| --- | --- |
| DELETE | 默认模式。在该模式下，在事务结束时，日志文件将被删除。 |
| TRUNCATE | 日志文件被阶段为零字节长度。 |
| PERSIST | 日志文件被留在原地，但头部被重写，表明日志不再有效。 |
| MEMORY | 日志记录保留在内存中，而不是磁盘上。 |
| OFF | 不保留任何日志记录。 |

# max_page_count Pragma

**max_page_count** Pragma 为数据库获取或设置允许的最大页数。语法如下：

```
PRAGMA [database.]max_page_count;
PRAGMA [database.]max_page_count = max_page;
```

默认值是 1,073,741,823，这是一个千兆的页面，即如果默认 1 KB 的页面大小，那么数据库中增长起来的一个兆字节。

## page_count Pragma

**page_count** Pragma 返回当前数据库中的网页数量。语法如下：

```
PRAGMA [database.]page_count;
```

数据库文件的大小应该是 page_count * page_size。

## page_size Pragma

**page_size** Pragma 获取或设置数据库页面的大小。语法如下：

```
PRAGMA [database.]page_size;
PRAGMA [database.]page_size = bytes;
```

默认情况下，允许的尺寸是 512、1024、2048、4096、8192、16384、32768 字节。改变现有数据库页面大小的唯一方法就是设置页面大小，然后立即 VACUUM 该数据库。

## parser_trace Pragma

**parser_trace** Pragma 随着它解析 SQL 命令来控制打印的调试状态，语法如下：

```
PRAGMA parser_trace = [true|false];
```

默认情况下，它被设置为 false，但设置为 true 时则启用，此时 SQL 解析器会随着它解析 SQL 命令来打印出它的状态。

## recursive_triggers Pragma

**recursive_triggers** Pragma 获取或设置递归触发器功能。如果未启用递归触发器，一个触发动作将不会触发另一个触发。语法如下：

```
PRAGMA recursive_triggers;
PRAGMA recursive_triggers = [true|false];
```

## schema_version Pragma

**schema_version** Pragma 获取或设置存储在数据库头中的的架构版本值。语法如下：

```
PRAGMA [database.]schema_version;
PRAGMA [database.]schema_version = number;
```

这是一个 32 位有符号整数值，用来跟踪架构的变化。每当一个架构改变命令执行（比如 CREATE... 或 DROP...）时，这个值会递增。

## secure_delete Pragma

**secure_delete** Pragma 用来控制内容是如何从数据库中删除。语法如下：

```
PRAGMA secure_delete;
PRAGMA secure_delete = [true|false];
PRAGMA database.secure_delete;
PRAGMA database.secure_delete = [true|false];
```

安全删除标志的默认值通常是关闭的，但是这是可以通过 SQLITE_SECURE_DELETE 构建选项来改变的。

## sql_trace Pragma

**sql_trace** Pragma 用于把 SQL 跟踪结果转储到屏幕上。语法如下：

```
PRAGMA sql_trace;
PRAGMA sql_trace = [true|false];
```

SQLite 必须通过 SQLITE_DEBUG 指令来编译要引用的该 Pragma。

## synchronous Pragma

**synchronous** Pragma 获取或设置当前磁盘的同步模式，该模式控制积极的 SQLite 如何将数据写入物理存储。语法如下：

```
PRAGMA [database.]synchronous;
PRAGMA [database.]synchronous = mode;
```

SQLite 支持下列同步模式：

| Pragma 值 | 描述 |
| --- | --- |
| 0 或 OFF | 不进行同步。 |
| 1 或 NORMAL | 在关键的磁盘操作的每个序列后同步。 |
| 2 或 FULL | 在每个关键的磁盘操作后同步。 |

## temp_store Pragma

**temp_store** Pragma 获取或设置临时数据库文件所使用的存储模式。语法如下：

```
PRAGMA temp_store;
PRAGMA temp_store = mode;
```

SQLite 支持下列存储模式：

| Pragma 值 | 描述 |
| --- | --- |
| 0 或 DEFAULT | 默认使用编译时的模式。通常是 FILE。 |
| 1 或 FILE | 使用基于文件的存储。 |
| 2 或 MEMORY | 使用基于内存的存储。 |

## temp_store_directory Pragma

**temp_store_directory** Pragma 获取或设置用于临时数据库文件的位置。语法如下：

```
PRAGMA temp_store_directory;
PRAGMA temp_store_directory = 'directory_path';
```

## user_version Pragma

**user_version** Pragma 获取或设置存储在数据库头的用户自定义的版本值。语法如下：

```
PRAGMA [database.]user_version;
PRAGMA [database.]user_version = number;
```

这是一个 32 位的有符号整数值，可以由开发人员设置，用于版本跟踪的目的。

## writable_schema Pragma

**writable_schema** Pragma 获取或设置是否能够修改系统表。语法如下：

```
PRAGMA writable_schema;
PRAGMA writable_schema = [true|false];
```

如果设置了该 Pragma，则表以 sqlite_ 开始，可以创建和修改，包括 sqlite_master 表。使用该 Pragma 时要注意，因为它可能导致整个数据库损坏。

← SQLite Distinct 关键字 SQLite 约束 →

点我分享笔记