

libvirtualpg  
2.0.0-RC0

Generated by Doxygen 1.8.14



# Contents

- 1 Main Page** **1**
  - 1.1 Introduction ..... 1
  
- 2 Data Structure Index** **3**
  - 2.1 Data Structures ..... 3
  
- 3 File Index** **5**
  - 3.1 File List ..... 5
  
- 4 Data Structure Documentation** **7**
  - 4.1 virtualPQstruct Struct Reference ..... 7
    - 4.1.1 Detailed Description ..... 8
  
- 5 File Documentation** **9**
  - 5.1 virtualpg.h File Reference ..... 9
    - 5.1.1 Detailed Description ..... 10
    - 5.1.2 Typedef Documentation ..... 10
      - 5.1.2.1 virtualPQ ..... 10
      - 5.1.2.2 virtualPQptr ..... 11
    - 5.1.3 Function Documentation ..... 11
      - 5.1.3.1 virtualpg\_extension\_init() ..... 11
      - 5.1.3.2 virtualpg\_version() ..... 11
  
- Index** **13**



# Chapter 1

## Main Page

### 1.1 Introduction

VirtualPG is an open source library implementing a Virtual Table extension to SQLite allowing to establish a connection to an external PostgreSQL/PostGIS Table or View.

Such a Virtual Table can then be handled using the usual SQLite's own SQL dialect (SELECT / INSERT / UPDATE / DELETE), more or less exactly as if it was a native Table.

Building and installing VirtualPG is straightforward:

```
./configure
make
make install
```

Linking VirtualPG to your own code is usually simple:

```
gcc my_program.c -o my_program -lvirtualpg -lsqlite3
```

On some systems you may have to provide a slightly more complex arrangement:

```
gcc -I/usr/local/include my_program.c -o my_program \
-L/usr/local/lib -lvirtualpg -lsqlite3
```

#### Note

VirtualPG obviously depends on **libpq** (the PostgreSQL own client library), but directly linking libpq (**-lpq**) isn't strictly required because VirtualPG can be initialized in such a way so to rely upon late binding instead of hard linking.

Please read the Tutorial for more detailed informations about this specific topic.

VirtualPG also provides pkg-config support, so you can also do:

```
gcc -I/usr/local/include my_program.c -o my_program `pkg-config --libs virtualpg`-lsqlite3
```

VirtualPG is licensed under the MPL tri-license terms: you are free to choose the best-fit license between:

- the MPL 1.1
- the GPL v2.0 or any subsequent version
- the LGPL v2.1 or any subsequent version

Enjoy, and happy coding



# Chapter 2

## Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

[virtualPQstruct](#)

virtualized libPQ methods . . . . . 7





# Chapter 3

## File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

[virtualpg.h](#)

Function declarations and constants for VirtualPG library . . . . . 9



## Chapter 4

# Data Structure Documentation

### 4.1 virtualPQstruct Struct Reference

virtualized libPQ methods

```
#include <virtualpq.h>
```

#### Data Fields

- void(\* [PQclear](#) )(PGresult \*res)  
*pointer to PQclear*
- PGconn \*([PQconnectdb](#) )(const char \*conninfo)  
*pointer to PQconnectdb*
- char \*([PQerrorMessage](#) )(const PGconn \*conn)  
*pointer to PQerrorMessage*
- PGresult \*([PQexec](#) )(PGconn \*conn, const char \*command)  
*pointer to PQexec*
- void(\* [PQfinish](#) )(PGconn \*conn)  
*pointer to PQfinish*
- int(\* [PQgetisnull](#) )(const PGresult \*res, int row\_number, int column\_number)  
*pointer to PQgetisnull*
- char \*([PQgetvalue](#) )(const PGresult \*res, int row\_number, int column\_number)  
*pointer to PQgetvalue*
- int(\* [PQlibVersion](#) )(void)  
*pointer to PQlibVersion*
- int(\* [PQnfields](#) )(const PGresult \*res)  
*pointer to PQnfields*
- int(\* [PQntuples](#) )(const PGresult \*res)  
*pointer to PQntuples*
- ExecStatusType(\* [PQresultStatus](#) )(const PGresult \*res)  
*pointer to PQresultStatus*
- ConnStatusType(\* [PQstatus](#) )(const PGconn \*conn)  
*pointer to PQstatus*

### 4.1.1 Detailed Description

virtualized libPQ methods

See also

[virtualPQptr](#)

The documentation for this struct was generated from the following file:

- [virtualpg.h](#)

# Chapter 5

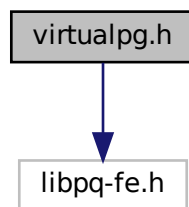
## File Documentation

### 5.1 virtualpg.h File Reference

Function declarations and constants for VirtualPG library

```
#include <libpq-fe.h>
```

Include dependency graph for virtualpg.h:



#### Data Structures

- struct [virtualPQstruct](#)

*virtualized libPQ methods*

#### Typedefs

- typedef struct [virtualPQstruct](#) [virtualPQ](#)

*virtualized libPQ methods*

- typedef [virtualPQ](#) \* [virtualPQptr](#)

*Typedef for virtual libPQ structure*

## Functions

- VIRTUALPG\_DECLARE void [vpgPQclear](#) (PGresult \*res)  
*virtual implementation for LibPQ - PQclear*
- VIRTUALPG\_DECLARE PGconn \* [vpgPQconnectdb](#) (const char \*conninfo)  
*virtual implementation for LibPQ - PQconnectdb*
- VIRTUALPG\_DECLARE char \* [vpgPQerrorMessage](#) (const PGconn \*conn)  
*virtual implementation for LibPQ - PQerrorMessage*
- VIRTUALPG\_DECLARE PGresult \* [vpgPQexec](#) (PGconn \*conn, const char \*command)  
*virtual implementation for LibPQ - PQexec*
- VIRTUALPG\_DECLARE void [vpgPQfinish](#) (PGconn \*conn)  
*virtual implementation for LibPQ - PQfinish*
- VIRTUALPG\_DECLARE int [vpgPQgetisnull](#) (const PGresult \*res, int row\_number, int column\_number)  
*virtual implementation for LibPQ - PQgetisnull*
- VIRTUALPG\_DECLARE char \* [vpgPQgetvalue](#) (const PGresult \*res, int row\_number, int column\_number)  
*virtual implementation for LibPQ - PQgetvalue*
- VIRTUALPG\_DECLARE int [vpgPQlibVersion](#) (void)  
*virtual implementation for LibPQ - PQlibVersion*
- VIRTUALPG\_DECLARE int [vpgPQnfields](#) (const PGresult \*res)  
*virtual implementation for LibPQ - PQnfields*
- VIRTUALPG\_DECLARE int [vpgPQntuples](#) (const PGresult \*res)  
*virtual implementation for LibPQ - PQntuples*
- VIRTUALPG\_DECLARE ExecStatusType [vpgPQresultStatus](#) (const PGresult \*res)  
*virtual implementation for LibPQ - PQresultStatus*
- VIRTUALPG\_DECLARE ConnStatusType [vpgPQstatus](#) (const PGconn \*conn)  
*virtual implementation for LibPQ - PQstatus*
- VIRTUALPG\_DECLARE const char \* [virtualpg\\_version](#) (void)  
  
*Return the current library version.*
- VIRTUALPG\_DECLARE int [virtualpg\\_extension\\_init](#) (sqlite3 \*db\_handle, [virtualPQptr](#) virtual\_api)

*Initializes libvirtualpg as an extension to SQLite*

### 5.1.1 Detailed Description

Function declarations and constants for VirtualPG library

### 5.1.2 Typedef Documentation

#### 5.1.2.1 virtualPQ

```
typedef struct virtualPQstruct virtualPQ
```

virtualized libPQ methods

See also

[virtualPQptr](#)

### 5.1.2.2 virtualPQptr

```
typedef virtualPQ* virtualPQptr
```

Typedef for virtual libPQ structure

See also

[virtualPQ](#)

## 5.1.3 Function Documentation

### 5.1.3.1 virtualpg\_extension\_init()

```
VIRTUALPG_DECLARE int virtualpg_extension_init (
    sqlite3 * db_handle,
    virtualPQptr virtual_api )
```

Initializes libvirtualpg as an extension to SQLite

#### Parameters

<i>db_handle</i>	pointer to the current DB connection.
<i>virtual_api</i>	pointer to a virtualPQ struct.

#### Returns

SQLITE\_OK will be returned on success, otherwise any appropriate error code on failure.

### 5.1.3.2 virtualpg\_version()

```
VIRTUALPG_DECLARE const char* virtualpg_version (
    void )
```

Return the current library version.

#### Returns

the version string.





# Index

- virtualPQptr
  - virtualpg.h, [10](#)
- virtualPQstruct, [7](#)
- virtualPQ
  - virtualpg.h, [10](#)
- virtualpg.h, [9](#)
  - virtualPQptr, [10](#)
  - virtualPQ, [10](#)
  - virtualpg\_extension\_init, [11](#)
  - virtualpg\_version, [11](#)
- virtualpg\_extension\_init
  - virtualpg.h, [11](#)
- virtualpg\_version
  - virtualpg.h, [11](#)