

Gaia DB

0.9

Generated by Doxygen 1.8.14

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	_db_ctx Struct Reference	5
3.1.1	Detailed Description	5
3.2	_star Struct Reference	5
3.2.1	Detailed Description	6
4	File Documentation	7
4.1	src/database_common.c File Reference	7
4.1.1	Detailed Description	8
4.1.2	Function Documentation	8
4.1.2.1	db_close()	8
4.1.2.2	db_get()	9
4.1.2.3	db_init()	9
4.1.2.4	db_insert()	10
4.1.2.5	log_error()	10
4.1.2.6	make_path()	10
4.2	src/database_common.h File Reference	11
4.2.1	Detailed Description	12
4.2.2	Function Documentation	12

4.2.2.1	db_close()	12
4.2.2.2	db_get()	13
4.2.2.3	db_init()	13
4.2.2.4	db_insert()	14
4.2.2.5	log_error()	14
4.3	src/gaia_db.c File Reference	14
4.3.1	Detailed Description	16
4.3.2	Function Documentation	16
4.3.2.1	gaia_close_cursor()	16
4.3.2.2	gaia_close_database()	16
4.3.2.3	gaia_cursor_get_star()	17
4.3.2.4	gaia_cursor_goto_star()	17
4.3.2.5	gaia_cursor_has_next()	18
4.3.2.6	gaia_delete_star()	18
4.3.2.7	gaia_get_star()	18
4.3.2.8	gaia_get_star_by_morton()	19
4.3.2.9	gaia_new_star()	19
4.3.2.10	gaia_open_cursor()	20
4.3.2.11	gaia_setup_database()	20
4.3.2.12	gaia_update_star_morton()	20
4.3.2.13	get_id_callback()	21
4.4	src/gaia_db.h File Reference	21
4.4.1	Detailed Description	23
4.4.2	Function Documentation	23
4.4.2.1	gaia_close_cursor()	23
4.4.2.2	gaia_close_database()	24
4.4.2.3	gaia_cursor_get_star()	24
4.4.2.4	gaia_cursor_goto_star()	24
4.4.2.5	gaia_cursor_has_next()	25
4.4.2.6	gaia_delete_star()	25
4.4.2.7	gaia_get_star()	26
4.4.2.8	gaia_get_star_by_morton()	26
4.4.2.9	gaia_new_star()	26
4.4.2.10	gaia_open_cursor()	27
4.4.2.11	gaia_setup_database()	27
4.4.2.12	gaia_update_star_morton()	28

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

_db_ctx	A small struct which holds pointers to databases and the directory they are in	5
_star	Star struct which holds basic data of a star	5

Chapter 2

File Index

2.1 File List

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

src/ database_common.c	
Helper functions for database interaction	7
src/ database_common.h	
Helper functions for database interaction	11
src/ gaia_db.c	
Implementation of the BerkeleyDB wrapper	14
src/ gaia_db.h	
Gaia DB wrapper	21

Chapter 3

Class Documentation

3.1 `_db_ctx` Struct Reference

A small struct which holds pointers to databases and the directory they are in.

```
#include <gaia_db.h>
```

Public Attributes

- DB * [dbp](#)
Handle to the primary database.
- DB * [sdbp](#)
Handle to the secondary database that holds indices for the morton codes.
- const char * [db_dir](#)
Home directory the databases are located in.

3.1.1 Detailed Description

A small struct which holds pointers to databases and the directory they are in.

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

- [src/gaia_db.h](#)

3.2 `_star` Struct Reference

Star struct which holds basic data of a star.

```
#include <gaia_db.h>
```

Public Attributes

- `u_int64_t morton_index`
Morton-code of the star in a 3d-grid.
- `u_int64_t id`
ID extracted from dataset.
- `double x`
X position star.
- `double y`
Y position star.
- `double z`
Z position star.
- `u_int32_t colour`
Colour of the star in hex converted to int.
- `float brightness`
Absolute magnitude of the star.

3.2.1 Detailed Description

Star struct which holds basic data of a star.

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

- `src/gaia_db.h`

Chapter 4

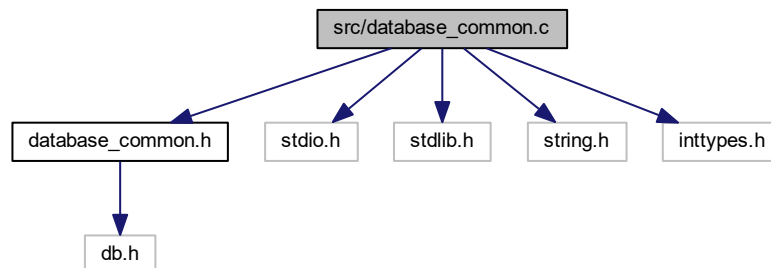
File Documentation

4.1 src/database_common.c File Reference

Helper functions for database interaction.

```
#include "database_common.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <inttypes.h>
```

Include dependency graph for database_common.c:



Functions

- char * [make_path](#) (const char *str1, const char *str2)
Form a path from the string to the directory and the name of the db. This needs to be freed. Done automatically by Berkeley if used to create db.
- void [log_error](#) (DB *dbp, int ret)
Small helper function for logging errors in the logfile.
- int [db_init](#) (DB **dbpp, const char *db_directory, const char *db_name, FILE *log_file, u_int32_t db_flags, DBTYPE db_type)
Initialize a database.
- int [db_close](#) (DB *dbp)

Close the database.

- int `db_insert` (DB *dbp, void *d_key, size_t s_key, void *d_data, size_t s_data)

Insert a value in the database.

- void * `db_get` (DB *dbp, void *d_key, int s_key)

Get an item from the database.

4.1.1 Detailed Description

Helper functions for database interaction.

Author

Danny Dorstijn

Version

0.9

Date

2019-01-23

Copyright

Copyright (c) 2019

4.1.2 Function Documentation

4.1.2.1 `db_close()`

```
int db_close (  
    DB * dbp )
```

Close the database.

Parameters

<code>dbp</code>	- Handle to the db to be closed
------------------	---------------------------------

Returns

int - Error code or 0 if all is fine

4.1.2.2 db_get()

```
void* db_get (
    DB * dbp,
    void * d_key,
    int s_key )
```

Get an item from the database.

Parameters

<i>dbp</i>	- Handle to the database
<i>d_key</i>	- Pointer to the key
<i>s_key</i>	- Size of the key

Returns

void* - Data of the record

4.1.2.3 db_init()

```
int db_init (
    DB ** dbpp,
    const char * db_directory,
    const char * db_name,
    FILE * log_file,
    u_int32_t db_flags,
    DBTYPE db_type )
```

Initialize a database.

Parameters

<i>dbpp</i>	- A pointer to a handle for the new db
<i>db_directory</i>	- The directory where to place the db
<i>db_name</i>	- Name of the database
<i>log_file</i>	- Log file to print all errors in
<i>db_flags</i>	- Flags for creating a db
<i>db_type</i>	- Type of the db (BTree, Heap, Queue, etc.)

Returns

int - Error code or 0 if all is fine

4.1.2.4 db_insert()

```
int db_insert (
    DB * dbp,
    void * d_key,
    size_t s_key,
    void * d_data,
    size_t s_data )
```

Insert a value in the database.

Parameters

<i>dbp</i>	- Handle to the db
<i>d_key</i>	- Pointer to the data of the key
<i>s_key</i>	- Size of the key (using sizeof)
<i>d_data</i>	- Pointer to the data
<i>s_data</i>	- Sizeof the corresponding data (using sizeof)

Returns

int - Error code or 0 if all is fine

4.1.2.5 log_error()

```
void log_error (
    DB * dbp,
    int ret )
```

Small helper function for logging errors in the logfile.

Parameters

<i>dbp</i>	- Handle to the database
<i>ret</i>	- The error code to be printed

4.1.2.6 make_path()

```
char* make_path (
    const char * str1,
    const char * str2 )
```

Form a path from the string to the directory and the name of the db. This needs to be freed. Done automatically by Berkeley if used to create db.

Parameters

<i>str1</i>	- First string
<i>str2</i>	- Second string

Returns

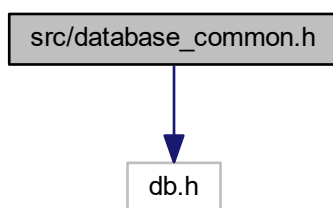
char* - Both strings combined

4.2 src/database_common.h File Reference

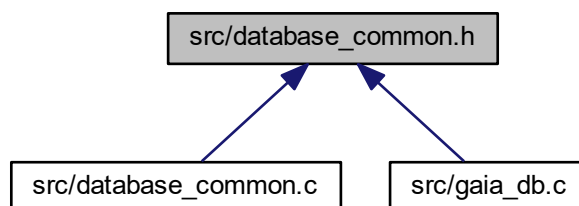
Helper functions for database interaction.

```
#include "db.h"
```

Include dependency graph for database_common.h:



This graph shows which files directly or indirectly include this file:



Functions

- void `log_error` (DB *dbp, int ret)
Small helper function for logging errors in the logfile.
- int `db_init` (DB **dbpp, const char *db_directory, const char *db_name, FILE *log_file, u_int32_t db_flags, DBTYPE db_type)
Initialize a database.
- int `db_close` (DB *dbp)
Close the database.
- int `db_insert` (DB *dbp, void *d_key, size_t s_key, void *d_data, size_t s_data)
Insert a value in the database.
- void * `db_get` (DB *dbp, void *d_key, int s_key)
Get an item from the database.

4.2.1 Detailed Description

Helper functions for database interaction.

Author

Danny Dorstijn

Version

0.9

Date

2019-01-23

Copyright

Copyright (c) 2019

4.2.2 Function Documentation

4.2.2.1 `db_close()`

```
int db_close (
    DB * dbp )
```

Close the database.

Parameters

<code>dbp</code>	- Handle to the db to be closed
------------------	---------------------------------

Returns

int - Error code or 0 if all is fine

4.2.2.2 db_get()

```
void* db_get (
    DB * dbp,
    void * d_key,
    int s_key )
```

Get an item from the database.

Parameters

<i>dbp</i>	- Handle to the database
<i>d_key</i>	- Pointer to the key
<i>s_key</i>	- Size of the key

Returns

void* - Data of the record

4.2.2.3 db_init()

```
int db_init (
    DB ** dbpp,
    const char * db_directory,
    const char * db_name,
    FILE * log_file,
    u_int32_t db_flags,
    DBTYPE db_type )
```

Initialize a database.

Parameters

<i>dbpp</i>	- A pointer to a handle for the new db
<i>db_directory</i>	- The directory where to place the db
<i>db_name</i>	- Name of the database
<i>log_file</i>	- Log file to print all errors in
<i>db_flags</i>	- Flags for creating a db
<i>db_type</i>	- Type of the db (BTree, Heap, Queue, etc.)

Returns

int - Error code or 0 if all is fine

4.2.2.4 db_insert()

```
int db_insert (
    DB * dbp,
    void * d_key,
    size_t s_key,
    void * d_data,
    size_t s_data )
```

Insert a value in the database.

Parameters

<i>dbp</i>	- Handle to the db
<i>d_key</i>	- Pointer to the data of the key
<i>s_key</i>	- Size of the key (using sizeof)
<i>d_data</i>	- Pointer to the data
<i>s_data</i>	- Sizeof the corresponding data (using sizeof)

Returns

int - Error code or 0 if all is fine

4.2.2.5 log_error()

```
void log_error (
    DB * dbp,
    int ret )
```

Small helper function for logging errors in the logfile.

Parameters

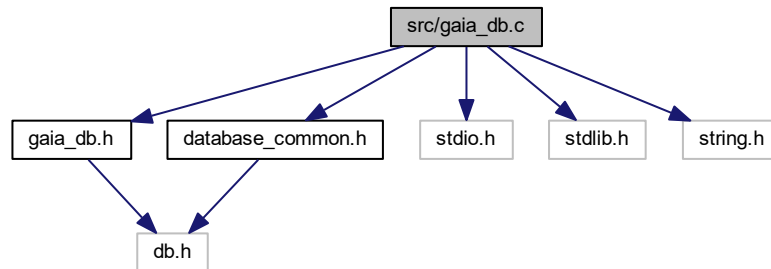
<i>dbp</i>	- Handle to the database
<i>ret</i>	- The error code to be printed

4.3 src/gaia_db.c File Reference

Implementation of the BerkeleyDB wrapper.

```
#include "gaia_db.h"
#include "database_common.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for gaia_db.c:



Functions

- `int get_id_callback (DB *dbp, const DBT *pkey, const DBT *pdata, DBT *skey)`
Callback used by the db for creating indices for the morton codes.
- `DB_CTX * gaia_setup_database (const char *directory)`
Setup the databases for GAIA.
- `int gaia_close_database (DB_CTX *ctx)`
Close the databases. Writes the db's to a file.
- `int gaia_new_star (DB *dbp, u_int64_t id, double x, double y, double z, u_int32_t colour, float brightness, u_int64_t morton_index)`
- `SStar * gaia_get_star (DB *dbp, u_int64_t id)`
Get a star from the db based on the ID of the star.
- `SStar * gaia_get_star_by_morton (DB *sdbp, u_int64_t index)`
Get the star based on it's morton code.
- `DBC * gaia_open_cursor (DB *dbp)`
Get a new cursor to iterate over the database.
- `int gaia_close_cursor (DBC *dbcp)`
Close the cursor after you are done using it.
- `SStar * gaia_cursor_get_star (DBC *dbcp)`
Get the star the cursor is pointing to.
- `char gaia_cursor_has_next (DBC *dbcp)`
Check if the cursor is on the last record and jump to that record.
- `int gaia_cursor_goto_star (DBC *dbcp, u_int64_t id)`
Set the cursor to a star with the id given.
- `int gaia_delete_star (DB *dbp, u_int64_t id)`
Search and remove a star from the db.
- `int gaia_update_star_morton (DB *dbp, u_int64_t id, u_int64_t morton_index)`
Update the morton code of the star.

4.3.1 Detailed Description

Implementation of the BerkeleyDB wrapper.

Author

Danny Dorstijn

Version

0.8

Date

2019-01-23

Copyright

Copyright (c) 2019

4.3.2 Function Documentation

4.3.2.1 `gaia_close_cursor()`

```
int gaia_close_cursor (
    DBC * dbcp )
```

Close the cursor after you are done using it.

Parameters

<code><i>dbcp</i></code>	- Handle to the database
--------------------------	--------------------------

Returns

int - Error code or 0 if all is fine

4.3.2.2 `gaia_close_database()`

```
int gaia_close_database (
    DB_CTX * ctx )
```

Close the databases. Writes the db's to a file.

Parameters

<i>ctx</i>	- The context with both the database handles
------------	--

Returns

int - Returns err code or 0 if all was fine

4.3.2.3 gaia_cursor_get_star()

```
SStar* gaia_cursor_get_star (  
    DBC * dbcp )
```

Get the star the cursor is pointing to.

Parameters

<i>dbcp</i>	- Handle to the database
-------------	--------------------------

Returns

SStar* - The star the cursor is pointing to

4.3.2.4 gaia_cursor_goto_star()

```
int gaia_cursor_goto_star (  
    DBC * dbcp,  
    u_int64_t id )
```

Set the cursor to a star with the id given.

Parameters

<i>dbcp</i>	- Handle to the database
<i>id</i>	- ID to jump to

Returns

int - Error code or 0 if all is fine

4.3.2.5 `gaia_cursor_has_next()`

```
char gaia_cursor_has_next (
    DBC * dbcp )
```

Check if the cursor is on the last record and jump to that record.

Parameters

<i>dbcp</i>	- Handle to the database
-------------	--------------------------

Returns

char - 1 if jump is succesful. Else 0

4.3.2.6 `gaia_delete_star()`

```
int gaia_delete_star (
    DB * dbp,
    u_int64_t id )
```

Search and remove a star from the db.

Parameters

<i>dbp</i>	- Handle for the primary database
<i>id</i>	- The id of the star

Returns

int - Error code or 0 if all is fine

4.3.2.7 `gaia_get_star()`

```
SStar* gaia_get_star (
    DB * dbp,
    u_int64_t id )
```

Get a star from the db based on the ID of the star.

Parameters

<i>dbp</i>	- Handle for the primary database
<i>id</i>	- ID of the star you are looking for

Returns

SStar* - Star object with all the data corresponding to the id given

4.3.2.8 gaia_get_star_by_morton()

```
SStar* gaia_get_star_by_morton (
    DB * sdbp,
    u_int64_t index )
```

Get the star based on it's morton code.

Parameters

<i>sdbp</i>	- Handle for the secondary database with morton code indices
<i>index</i>	- The morton code index

Returns

SStar* - The star data corresponding to the index given

4.3.2.9 gaia_new_star()

```
int gaia_new_star (
    DB * dbp,
    u_int64_t id,
    double x,
    double y,
    double z,
    u_int32_t colour,
    float brightness,
    u_int64_t morton_index )
```

Parameters

<i>dbp</i>	- Handle for the primary db
<i>id</i>	- ID of the star extracted from the uuid in the gaia dataset
<i>x</i>	- X position of the star
<i>y</i>	- Y position of the star
<i>z</i>	- Z position of the star
<i>colour</i>	- The colour of the star
<i>brightness</i>	- The brightness of the star (apparent magnitude)
<i>morton_index</i>	- The morton index of the star. Leave 0 if unsure

Returns

int - Error code or 0 if fine

4.3.2.10 gaia_open_cursor()

```
DBC* gaia_open_cursor (
    DB * dbp )
```

Get a new cursor to iterate over the database.

Parameters

<i>dbp</i>	- Handle to the database
------------	--------------------------

Returns

DBC* - The cursor

4.3.2.11 gaia_setup_database()

```
DB_CTX* gaia_setup_database (
    const char * directory )
```

Setup the databases for GAIA.

Parameters

<i>directory</i>	- The base directory to put the databases in
------------------	--

Returns

DB_CTX* - A helper to manage the databases

4.3.2.12 gaia_update_star_morton()

```
int gaia_update_star_morton (
    DB * dbp,
    u_int64_t id,
    u_int64_t morton_index )
```

Update the morton code of the star.

Parameters

<i>dbp</i>	- Handle of the primary database
<i>id</i>	- The id of the star
<i>morton_index</i>	- The new morton index

Returns

int - Error code or 0 if all is fine

4.3.2.13 get_id_callback()

```
int get_id_callback (
    DB * dbp,
    const DBT * pkey,
    const DBT * pdata,
    DBT * skey )
```

Callback used by the db for creating indices for the morton codes.

Parameters

<i>dbp</i>	- Handle for dbp (unused)
<i>pkey</i>	- Handle for the key of the main db (unused)
<i>pdata</i>	- Handle for the data of the main db. Used to extract morton idx
<i>skey</i>	- Handle for the secondary key. This is what we put in the db

Returns

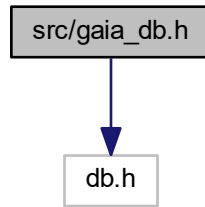
int - Returns 0 to signal all is fine

4.4 src/gaia_db.h File Reference

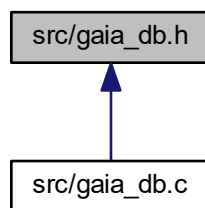
Gaia DB wrapper.

```
#include "db.h"
```

Include dependency graph for `gaia_db.h`:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_star`](#)

Star struct which holds basic data of a star.

- [struct `_db_ctx`](#)

A small struct which holds pointers to databases and the directory they are in.

Typedefs

- [typedef struct `_star` `SStar`](#)

Star struct which holds basic data of a star.

- [typedef struct `_db_ctx` `DB_CTX`](#)

A small struct which holds pointers to databases and the directory they are in.

Functions

- [DB_CTX](#) *GAIADB_DLL [gaia_setup_database](#) (const char *directory)
Setup the databases for GAIA.
- int GAIADB_DLL [gaia_close_database](#) ([DB_CTX](#) *context)
Close the databases. Writes the db's to a file.
- int GAIADB_DLL [gaia_new_star](#) (DB *dbp, u_int64_t id, double x, double y, double z, u_int32_t colour, float brightness, u_int64_t morton_index)
- [SStar](#) *GAIADB_DLL [gaia_get_star](#) (DB *dbp, u_int64_t id)
Get a star from the db based on the ID of the star.
- [SStar](#) *GAIADB_DLL [gaia_get_star_by_morton](#) (DB *sdbp, u_int64_t index)
Get the star based on it's morton code.
- int GAIADB_DLL [gaia_delete_star](#) (DB *dbp, u_int64_t id)
Search and remove a star from the db.
- int GAIADB_DLL [gaia_update_star_morton](#) (DB *dbp, u_int64_t id, u_int64_t morton_index)
Update the morton code of the star.
- DBC *GAIADB_DLL [gaia_open_cursor](#) (DB *dbp)
Get a new cursor to iterate over the database.
- char GAIADB_DLL [gaia_cursor_has_next](#) (DBC *dbcp)
Check if the cursor is on the last record and jump to that record.
- [SStar](#) *GAIADB_DLL [gaia_cursor_get_star](#) (DBC *dbcp)
Get the star the cursor is pointing to.
- int GAIADB_DLL [gaia_cursor_goto_star](#) (DBC *dbcp, u_int64_t id)
Set the cursor to a star with the id given.
- int GAIADB_DLL [gaia_close_cursor](#) (DBC *dbcp)
Close the cursor after you are done using it.

4.4.1 Detailed Description

Gaia DB wrapper.

Author

Danny Dorstijn

Version

0.8

Date

2019-01-23

Copyright

Copyright (c) 2019

4.4.2 Function Documentation

4.4.2.1 [gaia_close_cursor\(\)](#)

```
int GAIADB_DLL gaia\_close\_cursor (
    DBC * dbcp )
```

Close the cursor after you are done using it.

Parameters

<i>dbcp</i>	- Handle to the database
-------------	--------------------------

Returns

int - Error code or 0 if all is fine

4.4.2.2 gaia_close_database()

```
int GAIADB_DLL gaia_close_database (
    DB_CTX * ctx )
```

Close the databases. Writes the db's to a file.

Parameters

<i>ctx</i>	- The context with both the database handles
------------	--

Returns

int - Returns err code or 0 if all was fine

4.4.2.3 gaia_cursor_get_star()

```
SStar* GAIADB_DLL gaia_cursor_get_star (
    DBC * dbcp )
```

Get the star the cursor is pointing to.

Parameters

<i>dbcp</i>	- Handle to the database
-------------	--------------------------

Returns

SStar* - The star the cursor is pointing to

4.4.2.4 gaia_cursor_goto_star()

```
int GAIADB_DLL gaia_cursor_goto_star (
    DBC * dbcp,
    u_int64_t id )
```

Set the cursor to a star with the id given.

Parameters

<i>dbcp</i>	- Handle to the database
<i>id</i>	- ID to jump to

Returns

int - Error code or 0 if all is fine

4.4.2.5 gaia_cursor_has_next()

```
char GAIADB_DLL gaia_cursor_has_next (
    DBC * dbcp )
```

Check if the cursor is on the last record and jump to that record.

Parameters

<i>dbcp</i>	- Handle to the database
-------------	--------------------------

Returns

char - 1 if jump is succesful. Else 0

4.4.2.6 gaia_delete_star()

```
int GAIADB_DLL gaia_delete_star (
    DB * dbp,
    u_int64_t id )
```

Search and remove a star from the db.

Parameters

<i>dbp</i>	- Handle for the primary database
<i>id</i>	- The id of the star

Returns

int - Error code or 0 if all is fine

4.4.2.7 `gaia_get_star()`

```
SStar* GAIADB_DLL gaia_get_star (
    DB * dbp,
    u_int64_t id )
```

Get a star from the db based on the ID of the star.

Parameters

<i>dbp</i>	- Handle for the primary database
<i>id</i>	- ID of the star you are looking for

Returns

SStar* - Star object with all the data corresponding to the id given

4.4.2.8 `gaia_get_star_by_morton()`

```
SStar* GAIADB_DLL gaia_get_star_by_morton (
    DB * sdbp,
    u_int64_t index )
```

Get the star based on it's morton code.

Parameters

<i>sdbp</i>	- Handle for the secondary database with morton code indices
<i>index</i>	- The morton code index

Returns

SStar* - The star data corresponding to the index given

4.4.2.9 `gaia_new_star()`

```
int GAIADB_DLL gaia_new_star (
    DB * dbp,
    u_int64_t id,
    double x,
    double y,
    double z,
    u_int32_t colour,
    float brightness,
    u_int64_t morton_index )
```

Parameters

<i>dbp</i>	- Handle for the primary db
<i>id</i>	- ID of the star extracted from the uuid in the gaia dataset
<i>x</i>	- X position of the star
<i>y</i>	- Y position of the star
<i>z</i>	- Z position of the star
<i>colour</i>	- The colour of the star
<i>brightness</i>	- The brightness of the star (apparent magnitude)
<i>morton_index</i>	- The morton index of the star. Leave 0 if unsure

Returns

int - Error code or 0 if fine

4.4.2.10 `gaia_open_cursor()`

```
DBC* GAIADB_DLL gaia_open_cursor (
    DB * dbp )
```

Get a new cursor to iterate over the database.

Parameters

<i>dbp</i>	- Handle to the database
------------	--------------------------

Returns

DBC* - The cursor

4.4.2.11 `gaia_setup_database()`

```
DB_CTX* GAIADB_DLL gaia_setup_database (
    const char * directory )
```

Setup the databases for GAIA.

Parameters

<i>directory</i>	- The base directory to put the databases in
------------------	--

Returns

DB_CTX* - A helper to manage the databases

4.4.2.12 gaia_update_star_morton()

```
int GAIADB_DLL gaia_update_star_morton (
    DB * dbp,
    u_int64_t id,
    u_int64_t morton_index )
```

Update the morton code of the star.

Parameters

<i>dbp</i>	- Handle of the primary database
<i>id</i>	- The id of the star
<i>morton_index</i>	- The new morton index

Returns

int - Error code or 0 if all is fine

Index

- [_db_ctx, 5](#)
- [_star, 5](#)
- database_common.c
 - [db_close, 8](#)
 - [db_get, 8](#)
 - [db_init, 9](#)
 - [db_insert, 9](#)
 - [log_error, 10](#)
 - [make_path, 10](#)
- database_common.h
 - [db_close, 12](#)
 - [db_get, 13](#)
 - [db_init, 13](#)
 - [db_insert, 14](#)
 - [log_error, 14](#)
- db_close
 - [database_common.c, 8](#)
 - [database_common.h, 12](#)
- db_get
 - [database_common.c, 8](#)
 - [database_common.h, 13](#)
- db_init
 - [database_common.c, 9](#)
 - [database_common.h, 13](#)
- db_insert
 - [database_common.c, 9](#)
 - [database_common.h, 14](#)
- gaia_close_cursor
 - [gaia_db.c, 16](#)
 - [gaia_db.h, 23](#)
- gaia_close_database
 - [gaia_db.c, 16](#)
 - [gaia_db.h, 24](#)
- gaia_cursor_get_star
 - [gaia_db.c, 17](#)
 - [gaia_db.h, 24](#)
- gaia_cursor_goto_star
 - [gaia_db.c, 17](#)
 - [gaia_db.h, 24](#)
- gaia_cursor_has_next
 - [gaia_db.c, 17](#)
 - [gaia_db.h, 25](#)
- gaia_db.c
 - [gaia_close_cursor, 16](#)
 - [gaia_close_database, 16](#)
 - [gaia_cursor_get_star, 17](#)
 - [gaia_cursor_goto_star, 17](#)
 - [gaia_cursor_has_next, 17](#)
 - [gaia_delete_star, 18](#)
 - [gaia_get_star, 18](#)
 - [gaia_get_star_by_morton, 19](#)
 - [gaia_new_star, 19](#)
 - [gaia_open_cursor, 20](#)
 - [gaia_setup_database, 20](#)
 - [gaia_update_star_morton, 20](#)
 - [get_id_callback, 21](#)
- gaia_db.h
 - [gaia_close_cursor, 23](#)
 - [gaia_close_database, 24](#)
 - [gaia_cursor_get_star, 24](#)
 - [gaia_cursor_goto_star, 24](#)
 - [gaia_cursor_has_next, 25](#)
 - [gaia_delete_star, 25](#)
 - [gaia_get_star, 25](#)
 - [gaia_get_star_by_morton, 26](#)
 - [gaia_new_star, 26](#)
 - [gaia_open_cursor, 27](#)
 - [gaia_setup_database, 27](#)
 - [gaia_update_star_morton, 28](#)
- gaia_delete_star
 - [gaia_db.c, 18](#)
 - [gaia_db.h, 25](#)
- gaia_get_star
 - [gaia_db.c, 18](#)
 - [gaia_db.h, 25](#)
- gaia_get_star_by_morton
 - [gaia_db.c, 19](#)
 - [gaia_db.h, 26](#)
- gaia_new_star
 - [gaia_db.c, 19](#)
 - [gaia_db.h, 26](#)
- gaia_open_cursor
 - [gaia_db.c, 20](#)
 - [gaia_db.h, 27](#)
- gaia_setup_database
 - [gaia_db.c, 20](#)
 - [gaia_db.h, 27](#)
- gaia_update_star_morton
 - [gaia_db.c, 20](#)
 - [gaia_db.h, 28](#)
- get_id_callback
 - [gaia_db.c, 21](#)
- log_error
 - [database_common.c, 10](#)
 - [database_common.h, 14](#)
- make_path

database_common.c, [10](#)
src/database_common.c, [7](#)
src/database_common.h, [11](#)
src/gaia_db.c, [14](#)
src/gaia_db.h, [21](#)