

API Documentation

API Documentation

November 21, 2016

Contents

Contents	1
1 Module application	3
1.1 Functions	3
2 Module btree	4
2.1 Class BTree	4
2.1.1 Methods	4
2.1.2 Class Variables	4
3 Module buffer	5
3.1 Variables	5
3.2 Class Buffer	5
3.2.1 Methods	5
3.2.2 Class Variables	5
4 Module config_datablock	6
4.1 Class ConfigDatablock	6
4.1.1 Methods	6
4.1.2 Class Variables	6
5 Module datablock	7
5.1 Class Datablock	7
5.1.1 Methods	7
5.1.2 Class Variables	7
6 Module datafile	8
6.1 Class Datafile	8
6.1.1 Methods	8
6.1.2 Class Variables	8
7 Module leaf_datablock	9
7.1 Class LeafDatablock	9
7.1.1 Methods	9
7.1.2 Class Variables	10
8 Module node_datablock	11
8.1 Class NodeDatablock	11

8.1.1	Methods	11
8.1.2	Class Variables	11
9	Module record	13
9.1	Functions	13
9.2	Class Record	13
9.2.1	Methods	13
9.2.2	Class Variables	13
10	Module rowid	14
10.1	Class Rowid	14
10.1.1	Methods	14
10.1.2	Class Variables	14
11	Module table	15
11.1	Class Table	15
11.1.1	Methods	15
11.1.2	Class Variables	15
12	Module table_datablock	16
12.1	Class TableDatablock	16
12.1.1	Methods	16
12.1.2	Class Variables	17
Index		18

1 Module application

1.1 Functions

<code>parse_input(cmd, table)</code>

<code>parse_insert(values, table)</code>
--

<code>parse_select(values, table)</code>
--

<code>parse_update(values, table)</code>
--

<code>parse_delete(value, table)</code>

<code>parse_btree(table)</code>

2 Module btree

2.1 Class BTree

2.1.1 Methods

<code>init(self)</code>

<code>pretty_print(self, curr_dblock=None)</code>

<code>has_key(self, key_value)</code>

<code>find_key(self, key_value)</code>
--

<code>insert(self, key_value, rowid, curr_dblock=None)</code>

<code>split_during_insert(self, splited_data, src_dblock, datablock_type)</code>
--

<code>update(self, key_value, rowid)</code>

<code>delete(self, key_value)</code>

<code>new_root(self, left_dblock, right_dblock)</code>
--

2.1.2 Class Variables

Name	Description
root	Value: attr.ib()
buffer	Value: attr.ib()

3 Module buffer

3.1 Variables

Name	Description
BUFFER_SIZE	Value: 256

3.2 Class Buffer

3.2.1 Methods

init (<i>cls, datafile</i>)
Creates a new Datablock in memory from a string of bytes
flush (<i>self</i>)
new_datablock (<i>self, datablock_type, address</i>)
get_datablock_in_buffer (<i>self, address</i>)
is_datablock_in_buffer (<i>self, address</i>)
get_datablock (<i>self, address</i>)
search_dblock_with_free_space (<i>self, free_space, datablock_type</i>)
Load in the buffer the first data block with the space needed
linear_search_record (<i>self, datablock_type, value, field=None, unique=False</i>)
Search datablocks to contain a specific value
get_datablock_free_space (<i>self, dblock, free_space, datablock_type</i>)
get_next_empty_datablock (<i>self, address=0</i>)

3.2.2 Class Variables

Name	Description
datafile	Value: attr.ib()

4 Module config_datablock

4.1 Class ConfigDatablock



4.1.1 Methods

get_data(self)

Convert header and records to bytes

Overrides: datablock.Datablock.get_data

records_size(self)

update_btree_root(self, new_addr)

from_bytes(cls, address, data=None, count_record=0)

Creates a new TableDatablock in memory from a string of bytes

Overrides: datablock.Datablock.from_bytes

unpack(count_record, data)

Inherited from datablock.Datablock(Section 5.1)

delete(), empty()

4.1.2 Class Variables

Name	Description
btree_root	Value: attr.ib(default= 16300)
<i>Inherited from datablock.Datablock (Section 5.1)</i>	
DATABLOCK_SIZE, address, count_record, deleted, type	

5 Module datablock

5.1 Class Datablock

5.1.1 Methods

<code>get_data(<i>self</i>)</code>

<code>delete(<i>self</i>)</code>

<code>empty(<i>self</i>)</code>

<code>from_bytes(<i>cls</i>, <i>address</i>, <i>data</i>, <i>count_record</i>)</code>

Creates a new Datablock in memory from a string of bytes
--

5.1.2 Class Variables

Name	Description
address	Value: <code>attr.ib()</code>
type	Value: <code>attr.ib(default= 0)</code>
count_record	Value: <code>attr.ib(default= 0)</code>
deleted	Value: <code>attr.ib(default= False)</code>
DATABLOCK_SIZE	Value: <code>2* 1024</code>

6 Module datafile

6.1 Class Datafile

6.1.1 Methods

```
create_new(self)
```

```
get_datablock(self, address)
```

```
write_datablock(self, dblock)
```

```
next_available_datablock(self, address=0)
```

```
datablocks(self)
```

```
node_datablocks(self)
```

```
new_datablock(self, datablock_type, address)
```

6.1.2 Class Variables

Name	Description
filename	Value: attr.ib()
filesize	Value: 32* 1024* 1024
NUM_DATABLOCKS	Value: filesize/ Datablock.DATABLOCK_SIZE

7 Module *leaf_datablock*

7.1 Class *LeafDatablock*



7.1.1 Methods

get_data(*self*)

Convert header and records to bytes Format: TypeCountHeaderRecords

Overrides: datablock.Datablock.get_data

from_bytes(*cls*, *address*, *data=None*, *count_record=0*)

Creates a new TableDatablock in memory from a string of bytes

Overrides: datablock.Datablock.from_bytes

unpack(*count_record*, *data*)

find_key(*self*, *key_value*)

free_space(*self*)

can_insert(*self*)

insert(*self*, *key_value*, *rowid*)

insert_and_split(*self*, *key_value*, *rowid*)

Insert and split leaf in the btree

update_rowid(*self*, *key_value*, *rowid*)

delete(*self*, *key_value*)

Overrides: datablock.Datablock.delete

update_data(*self*, *keys=[]*, *rowids=[]*)

Inherited from datablock.Datablock(Section 5.1)

`empty()`

7.1.2 Class Variables

Name	Description
<code>rowids</code>	Value: <code>attr.ib(default= [])</code>
<code>keys</code>	Value: <code>attr.ib(default= [])</code>
<i>Inherited from datablock.Datablock (Section 5.1)</i>	
DATABLOCK_SIZE, address, count_record, deleted, type	

8 Module `node_datablock`

8.1 Class `NodeDatablock`

`datablock.Datablock`  `node_datablock.NodeDatablock`

8.1.1 Methods

`get_data(self)`

Convert header and records to bytes Format: TypeCountHeaderRecords

Overrides: `datablock.Datablock.get_data`

`from_bytes(cls, address, data=None, count_record=0)`

Creates a new `TableDatablock` in memory from a string of bytes

Overrides: `datablock.Datablock.from_bytes`

`unpack(count_record, data)`

`find_key(self, key_value)`

`update_data(self, keys=[], nexts=[])`

`free_space(self)`

`can_insert(self)`

`insert(self, key_value, left_addr, right_addr)`

`insert_and_split(self, key_value, left_addr, right_addr)`

Insert and split node in the btree

Inherited from `datablock.Datablock`(Section 5.1)

`delete()`, `empty()`

8.1.2 Class Variables

Name	Description
nexts	Value: attr.ib(default= [])
keys	Value: attr.ib(default= [])
<i>Inherited from datablock.Datablock (Section 5.1)</i>	
DATABLOCK_SIZE, address, count_record, deleted, type	

9 Module record

9.1 Functions

```
len_less_than_200(instance, attribute, value)
```

9.2 Class Record

9.2.1 Methods

```
pack(self)
```

```
writable_size(self, str_obj)
```

```
size(self)
```

9.2.2 Class Variables

Name	Description
code	Value: attr.ib(validator=attr.validators.instance_of(int))
description	Value: attr.ib(validator=len_less_than_200)
rowid	Value: attr.ib(default= None)
deleted	Value: attr.ib(default= False)

10 Module rowid

10.1 Class Rowid

Rowid points to a record in a datablock

10.1.1 Methods

get_record(<i>self</i>)
Returns a Record from the Rowid
pack(<i>self</i>)

10.1.2 Class Variables

Name	Description
dblock	Value: attr.ib()
pos	Value: attr.ib(validator= attr.validators.instance_of(int))

11 Module table

11.1 Class Table

11.1.1 Methods

init (<i>cls, datafile</i>)

insert (<i>self, code, desc</i>)

Inserts code and desc into table

select_code (<i>self, code</i>)
--

Finds record with code Uses btree for index

select_desc (<i>self, desc</i>)
--

Finds record by description Can't use btree

update (<i>self, code, desc</i>)

Updates record code with new description desc Finds record through select_code()
--

delete (<i>self, code</i>)

Deletes record by code Finds record through select_code()

exit (<i>self</i>)

11.1.2 Class Variables

Name	Description
BTREE_ROOT_DEFAULT	Value: 16300
buffer	Value: attr.ib(validator=attr.validators.instance_of(Buffer))
btree	Value: attr.ib(default= None)

12 Module *table_datablock*

12.1 Class *TableDatablock*

datablock.Datablock  *table_datablock*.TableDatablock

12.1.1 Methods

get_data(*self*)

Convert header and records to bytes

Overrides: datablock.Datablock.get_data

save_record(*self*, *record*)

Saves a Record to the datablock

records_size(*self*)

free_contiguous_space(*self*, *space_needed*)

write_data(*self*, *record*, *position*=None)

update_record(*self*, *record*, *desc*)

delete_record(*self*, *record*)

search_by(*self*, *value*, *field*)

get_record_by_pos(*self*, *position*)

Get specific record by its position

from_bytes(*cls*, *address*, *data*=None, *count_record*=0)

Creates a new TableDatablock in memory from a string of bytes

Overrides: datablock.Datablock.from_bytes

unpack(*count_record*, *data*)

unpack_records (<i>record_str</i> , <i>header</i> , <i>address</i>)
--

Returns a list of Records included in the datablock

Inherited from datablock.Datablock(Section 5.1)

delete(), empty()

12.1.2 Class Variables

Name	Description
header	Value: attr.ib(default= [])
records	Value: attr.ib(default= [])
<i>Inherited from datablock.Datablock (Section 5.1)</i>	
DATABLOCK_SIZE, address, count_record, deleted, type	

Index

- application (*module*), 3
 - application.parse_btree (*function*), 3
 - application.parse_delete (*function*), 3
 - application.parse_input (*function*), 3
 - application.parse_insert (*function*), 3
 - application.parse_select (*function*), 3
 - application.parse_update (*function*), 3
- btree (*module*), 4
 - btree.BTree (*class*), 4
 - btree.BTree.delete (*method*), 4
 - btree.BTree.find_key (*method*), 4
 - btree.BTree.has_key (*method*), 4
 - btree.BTree.init (*method*), 4
 - btree.BTree.insert (*method*), 4
 - btree.BTree.new_root (*method*), 4
 - btree.BTree.pretty_print (*method*), 4
 - btree.BTree.split_during_insert (*method*), 4
 - btree.BTree.update (*method*), 4
- buffer (*module*), 5
 - buffer.Buffer (*class*), 5
 - buffer.Buffer.flush (*method*), 5
 - buffer.Buffer.get_datablock (*method*), 5
 - buffer.Buffer.get_datablock_free_space (*method*), 5
 - buffer.Buffer.get_datablock_in_buffer (*method*), 5
 - buffer.Buffer.get_next_empty_datablock (*method*), 5
 - buffer.Buffer.init (*class method*), 5
 - buffer.Buffer.is_datablock_in_buffer (*method*), 5
 - buffer.Buffer.linear_search_record (*method*), 5
 - buffer.Buffer.new_datablock (*method*), 5
 - buffer.Buffer.search_dblock_with_free_space (*method*), 5
- config_datablock (*module*), 6
 - config_datablock.ConfigDatablock (*class*), 6
 - config_datablock.ConfigDatablock.records_size (*method*), 6
 - config_datablock.ConfigDatablock.unpack (*static method*), 6
 - config_datablock.ConfigDatablock.update_btree_root (*method*), 6
- datablock (*module*), 7
 - datablock.Datablock (*class*), 7
 - datablock.Datablock.delete (*method*), 7
 - datablock.Datablock.empty (*method*), 7
 - datablock.Datablock.from_bytes (*class method*), 7
 - datablock.Datablock.get_data (*method*), 7
- datafile (*module*), 8
 - datafile.Datafile (*class*), 8
 - datafile.Datafile.create_new (*method*), 8
 - datafile.Datafile.datablocks (*method*), 8
 - datafile.Datafile.get_datablock (*method*), 8
 - datafile.Datafile.new_datablock (*method*), 8
 - datafile.Datafile.next_available_datablock (*method*), 8
 - datafile.Datafile.node_datablocks (*method*), 8
 - datafile.Datafile.write_datablock (*method*), 8
- leaf_datablock (*module*), 9–10
 - leaf_datablock.LeafDatablock (*class*), 9–10
 - leaf_datablock.LeafDatablock.can_insert (*method*), 9
 - leaf_datablock.LeafDatablock.find_key (*method*), 9
 - leaf_datablock.LeafDatablock.free_space (*method*), 9
 - leaf_datablock.LeafDatablock.insert (*method*), 9
 - leaf_datablock.LeafDatablock.insert_and_split (*method*), 9

- leaf_datablock.LeafDatablock.unpack (*static method*), 9
- leaf_datablock.LeafDatablock.update_data (*method*), 9
- leaf_datablock.LeafDatablock.update_rowid (*method*), 9
- node_datablock (*module*), 11–12
 - node_datablock.NodeDatablock (*class*), 11–12
 - node_datablock.NodeDatablock.can_insert (*method*), 11
 - node_datablock.NodeDatablock.find_key (*method*), 11
 - node_datablock.NodeDatablock.free_space (*method*), 11
 - node_datablock.NodeDatablock.insert (*method*), 11
 - node_datablock.NodeDatablock.insert_and_split (*method*), 11
 - node_datablock.NodeDatablock.unpack (*static method*), 11
 - node_datablock.NodeDatablock.update_data (*method*), 11
- record (*module*), 13
 - record.len_less_than_200 (*function*), 13
 - record.Record (*class*), 13
 - record.Record.pack (*method*), 13
 - record.Record.size (*method*), 13
 - record.Record.writeble_size (*method*), 13
- rowid (*module*), 14
 - rowid.Rowid (*class*), 14
 - rowid.Rowid.get_record (*method*), 14
 - rowid.Rowid.pack (*method*), 14
- table (*module*), 15
 - table.Table (*class*), 15
 - table.Table.delete (*method*), 15
 - table.Table.exit (*method*), 15
 - table.Table.init (*class method*), 15
 - table.Table.insert (*method*), 15
 - table.Table.select_code (*method*), 15
 - table.Table.select_desc (*method*), 15
 - table.Table.update (*method*), 15
 - table_datablock (*module*), 16–17
 - table_datablock.TableDatablock (*class*), 16–17
 - table_datablock.TableDatablock.delete_record (*method*), 16
 - table_datablock.TableDatablock.free_contiguous_space (*method*), 16
 - table_datablock.TableDatablock.get_record_by_pos (*method*), 16
 - table_datablock.TableDatablock.records_size (*method*), 16
 - table_datablock.TableDatablock.save_record (*method*), 16
 - table_datablock.TableDatablock.search_by (*method*), 16
 - table_datablock.TableDatablock.unpack (*static method*), 16
 - table_datablock.TableDatablock.unpack_records (*static method*), 16
 - table_datablock.TableDatablock.update_record (*method*), 16
 - table_datablock.TableDatablock.write_data (*method*), 16