

---

# **PROV 2 BigchainDB Documentation**

***Release 0.4.1***

**Martin Stoffers**

**Apr 09, 2018**



---

## Contents

---

|          |                                              |           |
|----------|----------------------------------------------|-----------|
| <b>1</b> | <b>PROV 2 BigchainDB</b>                     | <b>1</b>  |
| 1.1      | Introduction . . . . .                       | 1         |
| 1.2      | Installation . . . . .                       | 1         |
| 1.3      | Usage . . . . .                              | 2         |
| 1.4      | License . . . . .                            | 3         |
| <b>2</b> | <b>Development</b>                           | <b>5</b>  |
| 2.1      | Contribute . . . . .                         | 5         |
| 2.2      | Setup . . . . .                              | 5         |
| 2.3      | Execute tests . . . . .                      | 5         |
| 2.4      | Coverage report . . . . .                    | 5         |
| 2.5      | Compile documentation . . . . .              | 6         |
| <b>3</b> | <b>Changelog</b>                             | <b>7</b>  |
| 3.1      | Version 0.4.1 (2018-04-09) . . . . .         | 7         |
| 3.2      | Version 0.4.0 (2017-06-29) . . . . .         | 7         |
| 3.3      | Version 0.3.1 (2017-04-07) . . . . .         | 7         |
| 3.4      | Version 0.3.0 (2017-04-01) . . . . .         | 7         |
| 3.5      | Version 0.2.0 (2017-03-05) . . . . .         | 7         |
| 3.6      | Version 0.1.0 (2017-02-21) . . . . .         | 8         |
| <b>4</b> | <b>Testing Howto</b>                         | <b>9</b>  |
| 4.1      | 1. Setup your env . . . . .                  | 9         |
| 4.2      | 2. Start your BigchainDB test node . . . . . | 9         |
| 4.3      | 3. Setup environment variables . . . . .     | 9         |
| 4.4      | 4. Run your tests . . . . .                  | 10        |
| <b>5</b> | <b>prov2bigchaindb</b>                       | <b>11</b> |
| 5.1      | prov2bigchaindb package . . . . .            | 11        |
| <b>6</b> | <b>Indices and tables</b>                    | <b>19</b> |
|          | <b>Python Module Index</b>                   | <b>21</b> |



### 1.1 Introduction

This python module provides three different clients to store [W3C-PROV](#) documents into a federation of [BigchainDB](#) nodes. All clients are implemented with respect to the proposed concepts from the masters thesis [Trustworthy Provenance Recording using a blockchain-like database](#).

See full documentation at: [prov2bigchaindb.readthedocs.io](http://prov2bigchaindb.readthedocs.io)

#### 1.1.1 Software Requirements

- Python 3.5 or Python 3.6
- Python development libraries
- GCC and Make
- A local rethinkdb server

### 1.2 Installation

#### 1.2.1 PyPi

Install it by running:

```
pip install prov2bigchaindb
```

You can view [prov2bigchaindb](#) on PyPi's package index

#### 1.2.2 Source

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb
```

```
# Setup virtual environment
python -m venv env
source env/bin/activate

# Install dependencies and package into virtual enviroment
make setup
```

## 1.3 Usage

### 1.3.1 DocumentConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
doc_client = clients.DocumentConceptClient(account_id="ID", host="127.0.0.1",
↪port=9984)

# Store a document
tx_id = doc_client.save_document(prov_document)

# Retrieve a document
doc = doc_client.get_document(tx_id)
```

### 1.3.2 GraphConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
graph_client = clients.GraphConceptClient(host="127.0.0.1", port=9984)

# Store a document
tx_ids = graph_client.save_document(prov_document)

# Retrieve a document
doc = graph_client.get_document(tx_ids)
```

### 1.3.3 RoleConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
role_client = clients.RoleConceptClient(host="127.0.0.1", port=9984)

# Store a document
tx_ids = role_client.save_document(prov_document)

# Retrieve a document
doc = role_client.get_document(tx_ids)
```

## 1.4 License

See [LICENSE](#) file



### 2.1 Contribute

Please, fork the code on Github and develop your feature in a new branch split from the develop branch. Commit your code to the main project by sending a pull request onto the develop branch

- Issue Tracker: <https://github.com/DLR-SC/prov2bigchaindb/issues>
- Source Code: <https://github.com/DLR-SC/prov2bigchaindb>

### 2.2 Setup

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb

# Setup virtual environment
python -m venv env
source env/bin/activate

# Install dependencies
make dev-setup
```

### 2.3 Execute tests

```
make test
```

### 2.4 Coverage report

```
make coverage
```

## 2.5 Compile documentation

```
make docs
```

#### **3.1 Version 0.4.1 (2018-04-09)**

- Updated BigchainDB to 1.3.0
- Updated db-driver to 0.4.1
- Updated networkx to 2.1
- Updated prov to 1.5.2

#### **3.2 Version 0.4.0 (2017-06-29)**

- Updated bigchaindb components to 1.0.0rc1

#### **3.3 Version 0.3.1 (2017-04-07)**

- Added travis-ci support
- Updated documentation

#### **3.4 Version 0.3.0 (2017-04-01)**

- Support for the role-based concept

#### **3.5 Version 0.2.0 (2017-03-05)**

- Support for the graph-based concept
- Added unit tests

## 3.6 Version 0.1.0 (2017-02-21)

- Support for the document-based concept
- Added basic unit tests
- Intergation of Gitlab-CI

To run the test local follow the next steps

### 4.1 1. Setup your env

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb

# Setup virtual environment
python -m venv env
source env/bin/activate

# Install dependencies
make dev-setup
```

### 4.2 2. Start your BigchainDB test node

The tests require a running BigchainDB and RethinkDB instance.

```
make run
```

### 4.3 3. Setup environment variables

- BDB\_HOST: Default: 127.0.0.1
- BDB\_PORT: Default: 9984

If you like to connect to BigchainDB node, hosted on a different port and/or machine:

```
BDB_HOST=127.0.0.1  
BDB_HOST=9984
```

## 4.4 4. Run your tests

```
# Change to env  
source env/bin/activate  
#Start tests  
make test
```

## 5.1 prov2bigchaindb package

### 5.1.1 Subpackages

prov2bigchaindb.core package

Submodules

prov2bigchaindb.core.accounts module

```
class prov2bigchaindb.core.accounts.BaseAccount (account_id: str, store: prov2bigchaindb.core.local_stores.SqliteStore)
    Bases: object
    BigchainDB Base Account
    get_id () → str
        Get Account id
        Returns Internal id of Account
        Return type str
    get_public_key () → str
        Get public key
        Returns Public key of Account
        Return type str
class prov2bigchaindb.core.accounts.DocumentConceptAccount (account_id: str, store: prov2bigchaindb.core.local_stores.SqliteStore)
    Bases: prov2bigchaindb.core.accounts.BaseAccount
    BigchainDB Document Concept Account
    save_asset (asset: dict, bdb_connection: bigchaindb_driver.driver.BigchainDB) → str
        Write asset to BigchainDB
```

#### Parameters

- **asset** (*dict*) – Dictionary with asset data
- **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transaction Id

**Return type** *str*

```
class prov2bigchaindb.core.accounts.GraphConceptAccount (prov_element:
    prov.model.ProvElement,
    prov_relations:
    dict,      id_mapping:
    dict,      names-
    paces:    list, store:
    prov2bigchaindb.core.local_stores.SqliteStore
    =
    <prov2bigchaindb.core.local_stores.SqliteStore
    object>)
```

Bases: *prov2bigchaindb.core.accounts.BaseAccount*

BigchainDB Graph Concept Account

**get\_tx\_id**() → *str*

Get the tx\_id that describes the account in BigchainDB

**Returns** Transaction id of account

**Return type** *str*

**has\_relations\_with\_id**() → *bool*

Indicates whether an account has relation with ids or not :return: True if one or more relation does have ids :rtype: bool

**has\_relations\_without\_id**() → *bool*

Indicates whether an account has relation without ids or not :return: True if one or more relation does have ids :rtype: bool

**save\_relations\_with\_ids** (*bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → *list*

Writes all assets with relations (having ids) to BigchainDB

**Parameters** **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transactions ids of all relations

**Return type** *list*

**save\_relations\_without\_ids** (*bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) →

*list*  
Write all assets with relations to BigchainDB

**Parameters** **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transactions ids of all relations

**Return type** *list*

**save\_instance\_asset** (*bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → *str*

Write provenance describing the account to BigchainDB

**Parameters** **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transactions id of instance

**Return type** *str*

```
class prov2bigchaindb.core.accounts.RoleConceptAccount (agent:
    prov.model.ProvAgent,
    relations: list, elements:
    dict, id_mapping: dict,
    namespaces: list, store:
    prov2bigchaindb.core.local_stores.SqliteStore
    =
    <prov2bigchaindb.core.local_stores.SqliteStore
    object>)
```

Bases: `prov2bigchaindb.core.accounts.BaseAccount`

BigchainDB Graph Concept Account

**get\_tx\_id()** → str

Get the tx\_id that describes the account in BigchainDB

**Returns** Transaction id of account

**Return type** str

**save\_elements** (*bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → list

Writes all elements with assets to BigchainDB

**Parameters** **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transactions ids of all relations

**Return type** list

**save\_instance\_asset** (*bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → str

Write provenance describing the account to BigchainDB

**Parameters** **bdb\_connection** (*BigchainDB*) – Connection object for BigchainDB

**Returns** Transactions id of instance

**Return type** str

## prov2bigchaindb.core.clients module

```
class prov2bigchaindb.core.clients.BaseClient (host: str = '0.0.0.0', port:
    int = 9984, num_connections:
    int = 5, local_store:
    prov2bigchaindb.core.local_stores.SqliteStore
    = <prov2bigchaindb.core.local_stores.SqliteStore
    object>)
```

Bases: `object`

BigchainDB Base Client

**test\_transaction** (*tx: dict*) → bool

Validate a transaction against BigchainDB

**Parameters** **tx** (*dict*) – Transaction to test

**Returns** True or Exception

**Return type** bool

**save\_document** (*document: object*) → object

Abstract method to store a document

**Parameters** **document** (*object*) – Document to save

**Returns** id

**Return type** object

**get\_document** (*document\_id: object*) → prov.model.ProvDocument  
 Abstract method to retrieve a document

**Parameters** **document\_id** (*object*) – Document to save

**Return type** ProvDocument

```
class prov2bigchaindb.core.clients.DocumentConceptClient (account_id: str
                                                         = None, host:
                                                         str = '0.0.0.0',
                                                         port: int = 9984,
                                                         num_connections:
                                                         int = 1, local_store:
                                                         prov2bigchaindb.core.local_stores.SqliteStore
                                                         =
                                                         <prov2bigchaindb.core.local_stores.SqliteStore
                                                         object>)
```

Bases: *prov2bigchaindb.core.clients.BaseClient*

**save\_document** (*document: str*) → str  
 Write a document into BigchainDB

**Parameters** **document** (*str or bytes or ProvDocument*) – Document as JSON/XML/PROVN

**Returns** Transaction id of document

**Return type** str

**get\_document** (*tx\_id: str*) → prov.model.ProvDocument  
 Retrieve a document by transaction id from BigchainDB

**Parameters** **tx\_id** (*str*) – Transaction Id of Document

**Returns** Document as ProvDocument object

**Return type** ProvDocument

```
class prov2bigchaindb.core.clients.GraphConceptClient (host: str = '0.0.0.0',
                                                         port: int = 9984,
                                                         num_connections:
                                                         int = 5, local_store:
                                                         prov2bigchaindb.core.local_stores.SqliteStore
                                                         =
                                                         <prov2bigchaindb.core.local_stores.SqliteStore
                                                         object>)
```

Bases: *prov2bigchaindb.core.clients.BaseClient*

**static calculate\_account\_data** (*prov\_document: prov.model.ProvDocument*) → list  
 Transforms a ProvDocument into a tuple with ProvElement, list of ProvRelation and list of Namespaces

**Parameters** **prov\_document** – Document to transform

**Returns** List of tuples(element, relations, namespace)

**Return type** list

**save\_document** (*document: str*) → list  
 Write a document into BigchainDB

**Parameters** **document** (*str or BufferedReader or ProvDocument*) – Document as JSON/XML/PROVN

**Returns** List of transaction ids

**Return type** list

**get\_document** (*document\_tx\_ids: list*) → prov.model.ProvDocument

Retrieve a document by a list transaction ids from BigchainDB

**Parameters** **document\_tx\_ids** (*list*) – Transaction Ids of Document

**Returns** Document as ProvDocument object

**Return type** ProvDocument

```
class prov2bigchaindb.core.clients.RoleConceptClient (host: str = '0.0.0.0',  
                                                    port: int = 9984,  
                                                    num_connections:  
                                                    int = 5, local_store:  
                                                    prov2bigchaindb.core.local_stores.SqliteStore  
                                                    =  
                                                    <prov2bigchaindb.core.local_stores.SqliteStore  
                                                    object>)
```

Bases: *prov2bigchaindb.core.clients.BaseClient*

**static calculate\_account\_data** (*prov\_document: prov.model.ProvDocument*) → list

Transforms a ProvDocument into a list of tuples including: ProvAgent, list of ProvRelations from agent, list of ProvElements associated to ProvAgent, list of Namespaces

**Parameters** **prov\_document** – Document to transform

**Returns** List of tuples(ProvAgent, list(), list(), list())

**Return type** list

**save\_document** (*document: str*) → list

Write a document into BigchainDB

**Parameters** **document** (*str or BufferedReader or ProvDocument*) – Document as JSON/XML/PROVN

**Returns** List of transaction ids

**Return type** list

**get\_document** (*document\_tx\_ids: list*) → prov.model.ProvDocument

Returns a document by a list transaction ids from BigchainDB

**Parameters** **document\_tx\_ids** (*list*) – Transaction Ids of Document

**Returns** Document as ProvDocument object

**Return type** ProvDocument

## prov2bigchaindb.core.exceptions module

**exception** prov2bigchaindb.core.exceptions.**Prov2BigchainDBException**

Bases: *Exception*

**exception** prov2bigchaindb.core.exceptions.**CreateRecordException**

Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.**ParseException**

Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.**NoAccountFoundException**

Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.**AccountNotCreatedException**

Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.**NoRelationFoundException**

Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.TransactionIdNotFound  
 Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

**exception** prov2bigchaindb.core.exceptions.BlockIdNotFound  
 Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

## prov2bigchaindb.core.local\_stores module

**class** prov2bigchaindb.core.local\_stores.BaseStore (db\_name: str = None)  
 Bases: *object*

**clean\_tables** ()  
 Delete all entries from all tables (Used for unit tests)

**write\_account** (account\_id: str, public\_key: str, private\_key: str, tx\_id: str = None)  
 Writes a new account entry in to the table accounts

### Parameters

- **tx\_id** (str) – Transactions id
- **account\_id** (str) – Id of account
- **public\_key** (str) – Public key of account
- **private\_key** (str) – Private key of account

**get\_account** (account\_id: str) → tuple  
 Returns tuple of account from data by account\_id

**Parameters** **account\_id** (str) – Id of account

**Returns** Tuple with account\_id, public\_key, private\_key and tx\_id

**Return type** tuple

**write\_tx\_id** (account\_id: str, tx\_id: str)  
 Writes tx\_id for given account\_id

### Parameters

- **account\_id** (str) – Id of account
- **tx\_id** (str) – Transaction id, which represents the account in BigchainDB

**class** prov2bigchaindb.core.local\_stores.SqliteStore (db\_name: str = ':memory:')  
 Bases: *prov2bigchaindb.core.local\_stores.BaseStore*

**clean\_tables** ()  
 Delete all entries from all tables (Used for unit tests)

**write\_account** (account\_id: str, public\_key: str, private\_key: str, tx\_id: str = None)  
 Writes a new account entry in to the table accounts

### Parameters

- **tx\_id** (str) – Transactions id
- **account\_id** (str) – Id of account
- **public\_key** (str) – Public key of account
- **private\_key** (str) – Private key of account

**get\_account** (account\_id: str) → tuple  
 Returns tuple of account from data by account\_id

**Parameters** **account\_id** (str) – Id of account

**Returns** Tuple with account\_id, public\_key, private\_key and tx\_id

**Return type** `tuple`

**write\_tx\_id**(*account\_id: str, tx\_id: str*)

Writes tx\_id for given account\_id

**Parameters**

- **account\_id**(*str*) – Id of account
- **tx\_id**(*str*) – Transaction id, which represents the account in BigchainDB

## prov2bigchaindb.core.utils module

`prov2bigchaindb.core.utils.to_prov_document`(*content: str*) → `prov.model.ProvDocument`

Takes a string, bytes or ProvDocument as argument and return a ProvDocument The strings or bytes can contain JSON or XML representations of PROV

**Parameters** **content** – String or BufferedReader or ProvDocument

**Returns** `ProvDocument`

**Return type** `ProvDocument`

`prov2bigchaindb.core.utils.wait_until_valid`(*tx\_id: str, bdb\_connection: bigchaindb\_driver.driver.BigchainDB*)

Waits until a transaction is valid in BigchainDB

**Parameters**

- **tx\_id**(*str*) – Id of transaction to wait on
- **bdb\_connection**(*BigchainDB*) – Connection object for BigchainDB

`prov2bigchaindb.core.utils.is_valid_tx`(*tx\_id: str, bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → `bool`

Checks once if a transaction is valid

**Parameters**

- **tx\_id**(*str*) – Id of transaction to check
- **bdb\_connection**(*BigchainDB*) – Connection object for BigchainDB

**Returns** `True` if valid

**Return type** `bool`

`prov2bigchaindb.core.utils.is_block_to_tx_valid`(*tx\_id: str, bdb\_connection: bigchaindb\_driver.driver.BigchainDB*) → `bool`

Checks if block with transaction is valid

**Parameters**

- **tx\_id**(*str*) – Id of transaction which should be included in the
- **bdb\_connection**(*BigchainDB*) – Connection object for BigchainDB

**Returns** `True` if transactions is in block and block is valid

**Return type** `bool`

## Module contents

### 5.1.2 Submodules

### 5.1.3 prov2bigchaindb.version module

### 5.1.4 Module contents

## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### p

- `prov2bigchaindb`, [18](#)
- `prov2bigchaindb.core`, [18](#)
- `prov2bigchaindb.core.accounts`, [11](#)
- `prov2bigchaindb.core.clients`, [13](#)
- `prov2bigchaindb.core.exceptions`, [15](#)
- `prov2bigchaindb.core.local_stores`, [16](#)
- `prov2bigchaindb.core.utils`, [17](#)
- `prov2bigchaindb.version`, [18](#)



## A

AccountNotCreatedException, 15

## B

BaseAccount (class in prov2bigchaindb.core.accounts), 11

BaseClient (class in prov2bigchaindb.core.clients), 13

BaseStore (class in prov2bigchaindb.core.local\_stores), 16

BlockIdNotFound, 16

## C

calculate\_account\_data()  
(prov2bigchaindb.core.clients.GraphConceptClient  
static method), 14

calculate\_account\_data()  
(prov2bigchaindb.core.clients.RoleConceptClient  
static method), 15

clean\_tables() (prov2bigchaindb.core.local\_stores.BaseStore  
method), 16

clean\_tables() (prov2bigchaindb.core.local\_stores.SQLiteStore  
method), 16

CreateRecordException, 15

## D

DocumentConceptAccount (class in  
prov2bigchaindb.core.accounts), 11

DocumentConceptClient (class in  
prov2bigchaindb.core.clients), 14

## G

get\_account() (prov2bigchaindb.core.local\_stores.BaseStore  
method), 16

get\_account() (prov2bigchaindb.core.local\_stores.SQLiteStore  
method), 16

get\_document() (prov2bigchaindb.core.clients.BaseClient  
method), 13

get\_document() (prov2bigchaindb.core.clients.DocumentConceptClient  
method), 14

get\_document() (prov2bigchaindb.core.clients.GraphConceptClient  
method), 14

get\_document() (prov2bigchaindb.core.clients.RoleConceptClient  
method), 15

get\_id() (prov2bigchaindb.core.accounts.BaseAccount  
method), 11

get\_public\_key() (prov2bigchaindb.core.accounts.BaseAccount  
method), 11

get\_tx\_id() (prov2bigchaindb.core.accounts.GraphConceptAccount  
method), 12

get\_tx\_id() (prov2bigchaindb.core.accounts.RoleConceptAccount  
method), 13

GraphConceptAccount (class in  
prov2bigchaindb.core.accounts), 12

GraphConceptClient (class in  
prov2bigchaindb.core.clients), 14

## H

has\_relations\_with\_id()  
(prov2bigchaindb.core.accounts.GraphConceptAccount  
method), 12

has\_relations\_without\_id()  
(prov2bigchaindb.core.accounts.GraphConceptAccount  
method), 12

is\_block\_to\_tx\_valid() (in module  
prov2bigchaindb.core.utils), 17

is\_valid\_tx() (in module prov2bigchaindb.core.utils),  
17

## N

NoAccountFoundException, 15

NoRelationFoundException, 15

## P

ParseException, 15

prov2bigchaindb (module), 18

prov2bigchaindb.core (module), 18

prov2bigchaindb.core.accounts (module), 11

prov2bigchaindb.core.clients (module), 13

prov2bigchaindb.core.exceptions (module), 15

prov2bigchaindb.core.local\_stores (module), 16

prov2bigchaindb.core.utils (module), 17

prov2bigchaindb.version (module), 18

Prov2BigchainDBException, 15

## R

RoleConceptAccount (class in prov2bigchaindb.core.accounts), 12

RoleConceptClient (class in prov2bigchaindb.core.clients), 15

## S

save\_asset() (prov2bigchaindb.core.accounts.DocumentConceptAccount method), 11

save\_document() (prov2bigchaindb.core.clients.BaseClient method), 13

save\_document() (prov2bigchaindb.core.clients.DocumentConceptClient method), 14

save\_document() (prov2bigchaindb.core.clients.GraphConceptClient method), 14

save\_document() (prov2bigchaindb.core.clients.RoleConceptClient method), 15

save\_elements() (prov2bigchaindb.core.accounts.RoleConceptAccount method), 13

save\_instance\_asset() (prov2bigchaindb.core.accounts.GraphConceptAccount method), 12

save\_instance\_asset() (prov2bigchaindb.core.accounts.RoleConceptAccount method), 13

save\_relations\_with\_ids() (prov2bigchaindb.core.accounts.GraphConceptAccount method), 12

save\_relations\_without\_ids() (prov2bigchaindb.core.accounts.GraphConceptAccount method), 12

SqliteStore (class in prov2bigchaindb.core.local\_stores), 16

## T

test\_transaction() (prov2bigchaindb.core.clients.BaseClient method), 13

to\_prov\_document() (in module prov2bigchaindb.core.utils), 17

TransactionIdNotFound, 15

## W

wait\_until\_valid() (in module prov2bigchaindb.core.utils), 17

write\_account() (prov2bigchaindb.core.local\_stores.BaseStore method), 16

write\_account() (prov2bigchaindb.core.local\_stores.SqliteStore method), 16

write\_tx\_id() (prov2bigchaindb.core.local\_stores.BaseStore method), 16

write\_tx\_id() (prov2bigchaindb.core.local\_stores.SqliteStore method), 17