demo_clex_importer

August 17, 2024

1 Demonstration of STIX-D's Clex Importer Tool

1.1 Introduction

Welcome to this demonstration of the Clex Importer tool. The Clex Importer is a utility designed to populate the `lexicon` table in the STIX-D Corpus Database with entries from the Attempto Controlled English (ACE) common lexicon. ACE is a controlled natural language, enabling precise language processing for applications that require unambiguous interpretation by both humans and machines.

In this demonstration, we will walk through the process of importing the ACE common lexicon into the database. The tool reads the Clex lexicon file, parses its content, and systematically imports the lexical entries into the database. By the end of this demonstration, the `lexicon` table will be populated with sample entries to support ACE-based natural language processing tasks.

1.2 Agenda

1. Use Case
2. Project Design
3. Code Interaction with Database
4. Test Cases
   - All Tests
   - Unit Tests
   - Integration Tests
   - End-to-End Tests
5. Code Execution
   - Command Line Interface (in notebook)
   - Web Interface (not in notebook)

1.3 1. Use

The STIX-D Use Case L1 involves seeding the `stixd_corpus.lexicon` database table with lexical entries from the ACE Common Lexicon (Clex) or similar files. An administrator provides a URI to the lexicon file, and the system connects to the local database via the `mysql_repository.py` module. For each line in the lexicon file, the system extracts relevant character strings to create a word tag and form, generates a SHA256 hash of these components, and checks for the hash in the `lexicon` table. If the hash exists, it links the existing entry with the source ID; if not, it creates a new entry. The system also imports additional arguments into appropriate fields and outputs summary information or error messages as necessary.

1.4 2. Project Design

1.4.1 Project Overview

The Clex Importer tool imports lexical entries from the Attempto Controlled English (ACE) lexicon file, stored as Prolog facts, into the `lexicon` table of the STIX-D MySQL database. This tool is accessible via the command line or a web form served by a Flask API, where users input a URL pointing to an ACE lexicon file. The system then parses each Prolog fact, maps it to the appropriate attributes in the `lexicon` table, and creates relevant entries in the `stix_objects` table (i.e., source documents) and the `obj_lex_jt` junction table.

1.4.2 OOP Principles in the Project

This project employs object-oriented programming (OOP) principles to create a modular, extensible, and maintainable system. Key OOP principles include:

- **Abstraction**: The project uses abstract classes and methods to define interfaces and enforce a common structure. For example, the `Repository` class defines abstract methods for database interactions, implemented by `MySQLRepository`.
- **Encapsulation**: Each class is responsible for a specific aspect of the project, encapsulating related data and behavior. For example, `ClexImporter` encapsulates the logic for importing Clex entries, while `MySQLRepository` encapsulates database interactions.
- **Inheritance**: The project uses inheritance to create a hierarchy of classes with shared behavior. For example, `MySQLRepository` inherits from `Repository` to reuse common database interaction methods.
- **Polymorphism**: The project uses polymorphism to allow different classes to be used interchangeably. For example, the `Repository` interface allows different types of repositories to be used with the `ClexImporter`.

1.4.3 Key Modules and Their OOP Design

The project consists of the following key modules, each designed using OOP principles:

- **`ClexImporter` Class in `clex_importer.py`**:
  - **Responsibility**: Manages the importation of Clex entries into the database.
  - **Attributes**:
    * `db_repo`: Represents the database repository where Clex entries will be stored.
    * `uri`: The location of the Clex file to be imported.
  - **Methods**:
    * `import_clex_entries()`: Imports Clex entries from the specified file into the database.
    * `parse_clex_entry()`: Parses a single Clex entry from the file.
    * `map_clex_entry_to_lexicon()`: Maps the parsed Clex entry to the `lexicon` table schema.
- **`MySQLRepository` Class in `mysql_repository.py`**:
  - **Responsibility**: Abstracts the database interactions for MySQL databases.
  - **Attributes**:
    * `connection`: Represents the connection to the MySQL database.
    * `table_name`: The name of the table in the database.
  - **Methods**:

        - ∗ `save_stix_object()`: Inserts metadata about the Clex import into the `stix_objects` table.
        - ∗ `save_entry()`: Inserts lexicon entries into the `lexicon` table.
        - ∗ `link_entry_with_stix()`: Links lexicon entries with STIX objects in the `obj_lex_jt` table.
        - ∗ `find_entry_by_id()`: Checks if a given entry already exists in the `lexicon` table.
- `generate_stix_uuid` **Method**:
    - **Responsibility**: Generates a UUID for each Clex entry based on its word tag and form.
    - **Methods**:
        - ∗ `generate_uuid()`: Generates a UUID for the Clex entry based on its word tag and form.

1.5 3. Code Interaction with the Database

1.5.1 Database Interaction via `clex_importer.py` and the `MySQLRepository` Class

The `clex_importer.py` module interacts with the STIX-D Corpus Database through the `MySQLRepository` class, which abstracts the complexities of SQL operations and provides a streamlined interface for database tasks. This interaction ensures that lexicon entries are accurately imported and managed within the database, leveraging both services and abstraction layers.

A. The `ClexImporter` Class as a Service Layer:

- **Purpose**: The `ClexImporter` class is responsible for reading and processing the ACE Common Lexicon (Clex) file and inserting or updating entries in the database.
- **Workflow**:
    - The process begins by reading the Clex file and parsing each entry to extract relevant data.
    - `ClexImporter` interacts with the `MySQLRepository` to insert new records or update existing ones based on the parsed data.

B. The `MySQLRepository` Class as an Abstraction Layer:

- **Purpose**: The `MySQLRepository` class abstracts the SQL operations needed to interact with the MySQL database, providing a clean interface for essential tasks like inserting records, querying data, and linking entries across tables.
- **Services Provided**:
    - **Inserting STIX Objects**: The `save_stix_object` method inserts metadata about the Clex import into the `stix_objects` table, helping track the provenance and context of the lexicon entries.
    - **Inserting Lexicon Entries**: The `save_entry` method inserts lexicon entries into the `lexicon` table. It accepts a dictionary of entry data and ensures it is correctly stored in the database.
    - **Linking Entries**: The `link_entry_with_stix` method associates lexicon entries with STIX objects by inserting records into the `obj_lex_jt` junction table, maintaining relationships between different data entities.
    - **Checking for Existing Entries**: The `find_entry_by_id` method checks the `lexicon` table to determine if an entry (identified by a unique hash) already exists, helping to avoid duplicates and maintain data integrity.

C. Abstract Layers and Their Benefits:

- **Separation of Concerns**: By utilizing the `MySQLRepository` class, the `clex_importer.py` module does not directly handle SQL queries or database connections. Instead, it relies on high-level methods provided by the repository, allowing for easier modification or extension of database interactions without altering the core business logic.
- **Reusability and Maintainability**: The abstraction provided by `MySQLRepository` facilitates the reuse of database interaction methods across different parts of the application, reducing code duplication and enhancing maintainability.
- **Error Handling**: The repository class encapsulates error handling for database operations. If a SQL operation fails, the class handles exceptions gracefully, enabling `ClexImporter` to focus on the overall import process rather than the intricacies of database management.

D. Example Workflow:

- **Inserting a New Lexicon Entry**:
  - `ClexImporter` processes a line from the Clex file, generating a unique hash for the entry.
  - It uses `find_entry_by_id` to check if the entry already exists in the `lexicon` table.
  - If the entry is new, `ClexImporter` calls `save_entry` to insert the data into the `lexicon` table.
  - The entry is then linked to the corresponding STIX object via the `link_entry_with_stix` method.

In summary, `clex_importer.py` efficiently manages database interactions through the `MySQLRepository` class, benefiting from the abstraction and services provided by the repository. This design enhances the code's modularity, maintainability, and clarity, making it easier to handle complex database operations in a structured and efficient manner.

1.6 4. Test Cases

This section provides an overview of the testing strategy employed in the project, which includes unit, integration, and end-to-end tests. These tests ensure the reliability and correctness of the code by validating individual components, interactions between modules, and the complete system workflow from front-end to back-end.

1.6.1 Setup Notebook Environment

Before running the tests, ensure that the necessary packages are installed and the required modules are imported. If running this notebook for the first time, uncomment and execute the provided code cell to install the necessary dependencies.

[ ]:

```
# %pip install -r ../demos/requirements.txt
```

1.6.2 Import Necessary Libraries & Set Global Variables

[ ]:

```
# Import Standard Libraries
import os, sys, pytest
# from IPython.display import IFrame, display

# Get the current working directory (CWD)
```

```
cwd = os.getcwd()
# Move up two levels to reach the stixd directory
stixd_path = os.path.abspath(os.path.join(cwd, '..', '..'))
# Append the stixd directory to the Python path
sys.path.append(stixd_path)

# Load Jupyter Notebook extensions
%load_ext sql

# Define Global Variables
TEST_DIR = os.path.join(os.getcwd(), '../tests')
VERBOSITY = '-q' # Quiet
TRACEBACK = '--tb=line' # One line
```

```
[ ]: # Connect to the database
     %sql mysql+mysqlconnector://your_username:your_password@localhost:3306/
       ↪stixd_corpus
```

1.6.3 All Test Cases

You can execute all the tests in the STIX-D project using the command below. This command will run every test case in the test directory, providing a comprehensive check of the entire system in just 30-60 seconds. This is an efficient way to ensure all components of the project function as expected, especially after significant code changes.

Note: The end-to-end (e2e) tests may occasionally fail due to intermittent issues in the notebook environment. If a test fails, try rerunning it. If the issue persists, restart the notebook kernel and run the test again.

```
[ ]: # Run all tests in the test directory (~30-60 seconds)
     pytest.main([TEST_DIR, VERBOSITY, TRACEBACK])
```

```
........
........
........
........
                                                    [100%]
32 passed in 46.44s
```

```
[ ]: <ExitCode.OK: 0>
```

1.6.4 Unit Tests

Test Case 1: `gen_clex_uuid` The test in `test_20_gen_clex_uuid.py` validates the `generate_stix_uuid` function, which generates STIX-compliant UUIDs based on input parameters. The test uses parameterized inputs and mocks network requests to ensure that UUIDs are generated correctly under various scenarios. This test ensures the correct format and behavior of UUID generation.

```
[ ]: # Run a specific test file in the test directory
     test_file = "test_20_gen_clex_uuid.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts
==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 2 items

..\tests\test_20_gen_clex_uuid.py::test_generate_uuid[4-x-stixd-clex-https:\raw.
githubusercontent.com\ciioprof0\stixd\03c934281777fecd3edb1d8622310bbf0839c17d\t
ests\test_clex.pl] PASSED [ 50%]
..\tests\test_20_gen_clex_uuid.py::test_generate_uuid[4-x-stixd-clex-https:\raw.
githubusercontent.com\Attempto\Clex\20960a5ce07776cb211a8cfb25dc8c81fcdf25e2\cle
x_lexicon.pl] PASSED [100%]

============================== 2 passed in 0.29s
==============================
```

```
[ ]: <ExitCode.OK: 0>
```

Test Case 2: `mysql_repo` The test in `test_30_mysql_repo.py` focuses on the `MySQLRepository` class, which manages interactions with the MySQL database. The test includes scenarios for saving and retrieving entries and uses mocking to simulate database operations. This test ensures that the repository's methods correctly manage database entries.

```
[ ]: # Run a specific test file in the test directory
     test_file = "test_30_mysql_repo.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts
==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 2 items
```

```
..\tests\test_30_mysql_repo.py::test_save_and_load_entry PASSED

[ 50%]
..\tests\test_30_mysql_repo.py::test_find_entry_by_id PASSED

[100%]

============================== 2 passed in 0.12s

==============================
```

```
[ ]: <ExitCode.OK: 0>
```

Test Case 3: `lexicon_manager` The test in `test_57_lexicon_manager.py` validates the `LexiconManager` class, responsible for managing lexicon entries in the database. The test covers creating, linking, and processing lexicon entries, using mocks to simulate database interactions. This test ensures that the `LexiconManager` performs its tasks effectively and reliably.

```
[ ]: # Run a specific test file in the test directory
     test_file = "test_57_lexicon_manager.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts

==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 3 items

..\tests\test_57_lexicon_manager.py::test_create_lexicon_entry
PASSED    [ 33%]
..\tests\test_57_lexicon_manager.py::test_link_lexicon_entry
PASSED      [ 66%]
..\tests\test_57_lexicon_manager.py::test_process_word PASSED

[100%]

============================== 3 passed in 0.10s

==============================
```

```
[ ]: <ExitCode.OK: 0>
```

Test Case 4: `clex_importer_local` The test in `test_70_clex_importer_local.py` focuses on the `ClexImporter` class's functionality, particularly the `import_clex_entries` method. The test verifies that the method correctly imports data into the `lexicon` table and ensures accuracy by comparing the database state before and after the import. This test confirms the correctness of the lexicon import process.

[ ]:

```
# Run a specific test file in the test directory (~ 10 seconds)
test_file = "test_70_clex_importer_local.py"
pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts

==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 4 items

..\tests\test_70_clex_importer_local.py::test_import_clex_entries[1-adv-fast-
fast-None-67e9b1c5cbd53045919deda792be49b18b41a09b3bd328f9cc406bb27d951f62]
PASSED [ 25%]
..\tests\test_70_clex_importer_local.py::test_import_clex_entries[19-noun_pl-
months-month-
neutr-6e7ab17fe3f242d10f360197f40646b443db6079d730e9d746c96824a2606336]
PASSED [ 50%]
..\tests\test_70_clex_importer_local.py::test_import_clex_entries[39-iv_finsg-
walks-walk-
None-f02be7a15dcd7cca79dc9b1c141991d479120352658c50030c7268da9372e6ff]
PASSED [ 75%]
..\tests\test_70_clex_importer_local.py::test_import_clex_entries[58-dv_pp-
succeeded-succeed-
as-8ee745975fad537905042b710e2f602f6c6bbe6c72f123b3596ce0b962f2b23f]
PASSED [100%]

============================= 4 passed in 10.91s

==============================
```

[ ]:

```
<ExitCode.OK: 0>
```

Test Case 5: `clex_importer_ci` The test in `test_75_clex_importer_ci.py` also targets the `ClexImporter` class, with a focus on its integration in a continuous integration (CI) environment. The test uses mocking to simulate external dependencies and verify that entries are correctly processed and stored. This test ensures the `ClexImporter` performs as expected under CI conditions.

[ ]:

```
# Run a specific test file in the test directory
test_file = "test_75_clex_importer_ci.py"
pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================== test session starts
==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 1 item

..\tests\test_75_clex_importer_ci.py::test_import_clex_entries
PASSED    [100%]

============================== 1 passed in 0.12s
==============================
```

```
[ ]: <ExitCode.OK: 0>
```

1.6.5 Integration Tests

Integration tests validate that different modules work together correctly. These tests simulate real user workflows to ensure that the system behaves as expected across module boundaries.

Test Case 6: api The test in `test_80_api.py` evaluates the `/import_clex` endpoint of the Flask API, ensuring that it correctly handles requests to import Clex entries. The test suite includes scenarios for successful imports, bad requests, and various error conditions, verifying that the API responds appropriately under different situations. This test ensures the robustness of the API endpoint.

```
[ ]: # Run a specific test file in the test directory
     test_file = "test_80_api.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================== test session starts
==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 5 items

..\tests\test_80_api.py::test_import_clex_success PASSED

[ 20%]
```

```
..\tests\test_80_api.py::test_import_clex_bad_request PASSED

[ 40%]
..\tests\test_80_api.py::test_import_clex_request_exception PASSED

[ 60%]
..\tests\test_80_api.py::test_import_clex_mysql_error PASSED

[ 80%]
..\tests\test_80_api.py::test_import_clex_system_error PASSED

[100%]

============================== 5 passed in 0.16s

==============================
```

```
[ ]: <ExitCode.OK: 0>
```

1.6.6 End-to-End Tests

End-to-End (E2E) tests simulate real user interactions with the entire system, testing the complete workflow from the user interface to the database. These tests ensure that the application functions correctly as a whole, replicating the experience of an actual user.

Test Case 7: `e2e_local` The E2E test in `test_90_e2e_local.py` simulates a complete user interaction with the Flask web application. It uses Selenium WebDriver to automate form submission for importing Clex entries and verifies the application's response. This test ensures that the integration between the front-end and back-end components works as expected.

Note: The E2E tests may occasionally fail due to the notebook environment. If a test fails, try rerunning it. If it fails again, restart the notebook kernel and rerun the test.

```
[ ]: # Run a specific test file in the test directory (~15 seconds)
     test_file = "test_90_e2e_local.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts

==============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 1 item

..\tests\test_90_e2e_local.py::test_form_submission PASSED

[100%]
```

```
============================= 1 passed in 15.72s

==============================
```

```
[ ]: <ExitCode.OK: 0>
```

Test Case 8: `e2e_ci` The E2E test in `test_95_e2e_ci.py` is designed to run in a continuous integration environment. Similar to the local E2E test, it automates the form submission process and verifies the application's response, ensuring the system works correctly from end to end. This test is crucial for validating the application in a CI pipeline.

Note: As with the local E2E test, this test may occasionally fail in the notebook environment. Restarting the kernel and rerunning the test can help resolve these issues.

```
[ ]: # Run a specific test file in the test directory (~15 seconds)
     test_file = "test_95_e2e_ci.py"
     pytest.main([os.path.join(TEST_DIR, test_file), "-v", "--tb=auto"])
```

```
============================= test session starts

=============================
platform win32 -- Python 3.12.4, pytest-8.3.2, pluggy-1.5.0 --
d:\OneDrive\Code\hltms\stixd\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: d:\OneDrive\Code\hltms\stixd
configfile: pytest.ini
plugins: anyio-4.4.0, mock-3.14.0
collecting … collected 1 item

..\tests\test_95_e2e_ci.py::test_form_submission PASSED

[100%]

============================= 1 passed in 16.45s

==============================
```

```
[ ]: <ExitCode.OK: 0>
```

```
[ ]:
```

1.7 5. Code

In this section, we will explore how to execute the Clex Importer tool using both the command line interface (within this notebook) and a web interface (externally). The CLI provides direct control over the tool, while the web interface offers a more accessible way to import lexicon entries via a form. Before proceeding, we need to reset the database to ensure a clean state.

1.7.1 Reset the Database

When running the code cell below to reset the database, you may encounter `Error: 1064 (42000)`. This error occurs because the MySQL `DELIMITER` command is not recognized by the `mysql.connector` library used in Python. Despite this error, the SQL script executes as intended. The error can be safely ignored.

[ ]:

```
# Reset the database to start with an empty database
%run ../app/reset_database.py
```

```
Error: 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near
'DELIMITER ;


-- Create procedure to check for prolog constraints (sp_check_prol' at line 1
```

1.7.2 Show Reset Database

After resetting the database, all tables should be empty. Let's verify the initial state of the database by running a query to select all entries from the three tables affected by the Clex Importer tool.

[ ]:

```
# Fetch table counts
lexicon_count = %sql SELECT COUNT(*) FROM lexicon;
stix_objects_count = %sql SELECT COUNT(*) FROM stix_objects;
obj_lex_jt_count = %sql SELECT COUNT(*) FROM obj_lex_jt;

# Display number of rows in each table
print(f"\nRows in 'stix_objects' table:  {stix_objects_count[0][0]}\n"
      f"Rows in 'lexicon' table:       {lexicon_count[0][0]}\n"
      f"Rows in 'obj_lex_jt' table:    {obj_lex_jt_count[0][0]}")
```

```
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.

Rows in 'stix_objects' table:  0
Rows in 'lexicon' table:       0
Rows in 'obj_lex_jt' table:    0
```

1.7.3 Execution from Command Line

The Clex Importer tool can be executed from the command line. The following command demonstrates how to run the Clex Importer tool from the command line:

```
`python clex_importer.py --uri <URL_TO_CLEX_FILE>`
```

- After a database reset, expect 62 total entries with 59 new and 3 existing entries

- Without a database reset, expect 62 total entries with 0 new and 62 existing entries.

```
[ ]: %run ../app/clex_importer.py "https://github.com/ciioprof0/stixd/raw/main/
      ↪lexicon/test_clex.pl"
```

```
Saved STIX object with ID: x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 67e9b1c5cbd53045919deda792be49b18b41a09b3bd328f9cc406bb27d951f62
for adv - fast
Inserted entry with lex_id: 1 into lexicon
Linking lex_id 1 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 1 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 38a31bf0527ff6fd23c6be74bfba58c46dbad709ce90b6d09b9a26f103a326b5
for adv_comp - faster
Inserted entry with lex_id: 2 into lexicon
Linking lex_id 2 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 2 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 55fee0f355e343b2c6a4d63b72a8ea8bcaa1a71698ada04e01533a8dc98fb4ee
for adv_sup - fastest
Inserted entry with lex_id: 3 into lexicon
Linking lex_id 3 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 3 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: b0a248290b9aa18bfbbbfd5367dc0cc0dc82a9e90dd83b88cce59361b8d67e8a
for adv - quickly
Inserted entry with lex_id: 4 into lexicon
Linking lex_id 4 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 4 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: bbe9bafa7a2a6e250fdf482a7c46217d7c63ccee917b3ae48324b61659c7e32d
for adj_itr - large
Inserted entry with lex_id: 5 into lexicon
Linking lex_id 5 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 5 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: bf10c6415fdedfef6bb41e276ee11b5411b9735e04279a725fd1e10f73efd5a3
for adj_itr_comp - larger
Inserted entry with lex_id: 6 into lexicon
Linking lex_id 6 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 6 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
```

```
Generated hash: 272e32e00264453eae65c42e85ebd4e63d2050652adf2a83e6251f58d44c1f80
for adj_itr_sup - largest
Inserted entry with lex_id: 7 into lexicon
Linking lex_id 7 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 7 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 0a071608bacedfe582720a7eac91a086c2fd4b8310886756596e55a19831f1b2
for adj_itr - expensive
Inserted entry with lex_id: 8 into lexicon
Linking lex_id 8 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 8 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 3185ec15fd8fee48b0ce5f3cf0702e7342b8a79e924b9560672d46f85b57cf0d
for adj_tr - valid-for
Inserted entry with lex_id: 9 into lexicon
Linking lex_id 9 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 9 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: bea4050ddc4a98586180049a5701670bfda43dfb33a9e9d439e09ea6658d8b49
for adj_tr - fond-of
Inserted entry with lex_id: 10 into lexicon
Linking lex_id 10 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 10 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: f9af59d37b5027585c5c367ff09c9ae6c9a9c2a572f0c2c6e31cb7a477803f70
for adj_tr_comp - fonder-of
Inserted entry with lex_id: 11 into lexicon
Linking lex_id 11 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 11 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 0593075d289bc5204293471f2179dcb02b688666ab3f0609dc0707038bda6c19
for adj_tr_sup - fondest-of
Inserted entry with lex_id: 12 into lexicon
Linking lex_id 12 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 12 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ad8d6c20f19057bc423efd798ab1b7969e8d16b84175a5e1c8d6327f301c3555
for adj_tr - pessimistic-about
Inserted entry with lex_id: 13 into lexicon
Linking lex_id 13 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 13 with stix_uuid x-stixd-clex--
```

```
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 81a4335048050be30cbe511f20aa06edd43edeace4e509c9dbe295d91f3d7c67
for noun_sg - woman
Inserted entry with lex_id: 14 into lexicon
Linking lex_id 14 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 14 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: d94e7754776de6a607bd1401e97542788b8a5bcdd770a6743029ba3cb9281e9b
for noun_pl - women
Inserted entry with lex_id: 15 into lexicon
Linking lex_id 15 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 15 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 3e750200b57ae141249f9a284aa994912f051b9cdaaabce0eab54d8f709786b4
for noun_sg - credit-card
Inserted entry with lex_id: 16 into lexicon
Linking lex_id 16 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 16 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ba1d673f62b614589363b57193e41fff0a55c16496de3390cd4bdebe05b66a5f
for noun_pl - credit-cards
Inserted entry with lex_id: 17 into lexicon
Linking lex_id 17 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 17 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 7dd815d8b271ce4a800d0709cc47e4542ae83c20580fad16f0ce0ac47ab825cb
for noun_sg - month
Inserted entry with lex_id: 18 into lexicon
Linking lex_id 18 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 18 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 6e7ab17fe3f242d10f360197f40646b443db6079d730e9d746c96824a2606336
for noun_pl - months
Inserted entry with lex_id: 19 into lexicon
Linking lex_id 19 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 19 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ab4b87413dcf986e3987394af66d8cd5b721d6e2e3414f0bf79b7180c496cffc
for noun_mass - water
Inserted entry with lex_id: 20 into lexicon
Linking lex_id 20 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
```

```
Successfully linked lex_id 20 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 1f47f35dd555019a4069f9ccbd73b6ca9aba50e65cba190c6c2234a0ec959550
for noun_mass - fear
Inserted entry with lex_id: 21 into lexicon
Linking lex_id 21 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 21 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 54eb36da1c2712a561a7f3d97ae4711be26e689a610a5b3bcc6cf8246b4bd328
for noun_mass - money
Inserted entry with lex_id: 22 into lexicon
Linking lex_id 22 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 22 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: f0f05c2c9d5fbbca1a49caaf36e02cabd3e8008b6ba2e06dd4f3ad5af3fc6778
for mn_sg - kg
Inserted entry with lex_id: 23 into lexicon
Linking lex_id 23 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 23 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 4062f60546af1df473dca3885c5678bceba6f846a393bde85c9f062dd7b3e4e9
for mn_pl - kg
Inserted entry with lex_id: 24 into lexicon
Linking lex_id 24 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 24 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 355a5ac057c3579401933c1edab57acd83c4fff5af52b235fe6a6487740e7266
for mn_sg - m
Inserted entry with lex_id: 25 into lexicon
Linking lex_id 25 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 25 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: bd00a278eb02fcda5e6953929b33fb7b722616288a41ffec7e161932b00a4f28
for mn_pl - m
Inserted entry with lex_id: 26 into lexicon
Linking lex_id 26 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 26 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 4dbd3783cd9aaba7e2c7b91cd8b7528fe090214cf231c438a2bbce6c469eae54
for mn_sg - °C
Inserted entry with lex_id: 27 into lexicon
Linking lex_id 27 with stix_object_id x-stixd-clex--
```

```
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 27 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 0b159623ff61b9c0c6dcbade8aecbcf5a3ba43a5c78a7353af9d4fc9a89d387b
for mn_pl - °C
Inserted entry with lex_id: 28 into lexicon
Linking lex_id 28 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 28 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: cdb8203c69d244b96d0efc358b21bb7ecdfedaa377fc2d22174d202d514b60bd
for pn_sg - John
Inserted entry with lex_id: 29 into lexicon
Linking lex_id 29 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 29 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: c362d3b6c8f1563be3031c9a27e83307499ef757668783df75d1c2ade5a8dcdb
for pn_sg - Nokia
Inserted entry with lex_id: 30 into lexicon
Linking lex_id 30 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 30 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: b79f2d1e7257a89546b59cb6384c316c6a67b9090e0eb79e4945ab5203c5fcca
for pndef_sg - Nile
Inserted entry with lex_id: 31 into lexicon
Linking lex_id 31 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 31 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 0ba13f88729a6598583fd0d63c2c946bbefb8e7084b49d1a6add9c13cd87615c
for pndef_pl - United-Nations
Inserted entry with lex_id: 32 into lexicon
Linking lex_id 32 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 32 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ab1d4ceef2dea8b6524e093ed881a9b373c7399fdda1238c6d6a50e45cb2e5c1
for pn_sg - Mona-Lisa
Inserted entry with lex_id: 33 into lexicon
Linking lex_id 33 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 33 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 553a63ea05f5be1335cdbaf4776e9fefe7b739709bde3afbc52d13f90d7f0747
for pndef_sg - Mona-Lisa
Inserted entry with lex_id: 34 into lexicon
```

```
Linking lex_id 34 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 34 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ac7502fa73fb93115da3fdc332c095b93db172fb3c9da5342cb1d4efea1dbfa4
for iv_finsg - waits
Inserted entry with lex_id: 35 into lexicon
Linking lex_id 35 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 35 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: efb99022f10de88cfa917fd5f2d22db31a264c47c446e39b745ecaf2104386d5
for iv_infpl - wait
Inserted entry with lex_id: 36 into lexicon
Linking lex_id 36 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 36 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 43b09b0fdca3918fdeffbdd00f9b9b1e36680a19e18a27b9ef837250aea760ff
for iv_finsg - goes-away
Inserted entry with lex_id: 37 into lexicon
Linking lex_id 37 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 37 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ecc9360327520149b50d67f39339a5340963d1f856c8e2379694e5758b18b90f
for iv_infpl - go-away
Inserted entry with lex_id: 38 into lexicon
Linking lex_id 38 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 38 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: f02be7a15dcd7cca79dc9b1c141991d479120352658c50030c7268da9372e6ff
for iv_finsg - walks
Inserted entry with lex_id: 39 into lexicon
Linking lex_id 39 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 39 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 31e757a6e9e9b7a3e8b7760ed46a8d8fac08fa4989bd1e2564178a466592fb0b
for iv_infpl - walk
Inserted entry with lex_id: 40 into lexicon
Linking lex_id 40 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 40 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ee65aeba7add7b5c2bec1c8453b46d0686c74980f5b66e971d8e0fdf4be2f339
for tv_finsg - knows
```

```
Inserted entry with lex_id: 41 into lexicon
Linking lex_id 41 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 41 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 422c3c3b73090cafb154a7af0c97a4a8b6526d0c081ecfd3c4852555ca497274
for tv_infpl - know
Inserted entry with lex_id: 42 into lexicon
Linking lex_id 42 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 42 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 77ca0e15adaec7eb9f1e2b265d8c4aaea36eb3233610413ec08cbd2cc9e75602
for tv_pp - known
Inserted entry with lex_id: 43 into lexicon
Linking lex_id 43 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 43 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: f7e5589db0c1cd8731249783f142502b70a9c4c81f2438a1a68d9a14466d7638
for tv_finsg - likes
Inserted entry with lex_id: 44 into lexicon
Linking lex_id 44 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 44 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 5662a9d8c3e47a4030ee3ce04f64b96d20f97050b43889b9c99a3d75bdcf2e87
for tv_infpl - like
Inserted entry with lex_id: 45 into lexicon
Linking lex_id 45 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 45 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: f4758877070e53c2c6289823abd379e18f73118cf9427ba32004de974786bd71
for tv_pp - liked
Inserted entry with lex_id: 46 into lexicon
Linking lex_id 46 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 46 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: ee6ebaf5ffc016337f2f65c9331f36a1fbe639b1f78ae0447d006d28d8056028
for tv_finsg - relates-to
Inserted entry with lex_id: 47 into lexicon
Linking lex_id 47 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 47 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 192eda123579168e6bd639ed1df16c5698e593c45a353c7dac8e4ba71a910a72
```

```
for tv_infpl - relate-to
Inserted entry with lex_id: 48 into lexicon
Linking lex_id 48 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 48 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 90837f23fb3398bbf7acd13a4734d872ad7561a56638f7d02aef58af5f950656
for tv_pp - related-to
Inserted entry with lex_id: 49 into lexicon
Linking lex_id 49 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 49 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 42f5f444d7d957ad84c5a633757577db59591b988579f53bd1f350baddaf9f03
for dv_finsg - shows
Inserted entry with lex_id: 50 into lexicon
Linking lex_id 50 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 50 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: d14f9f05e5a0b45180e21ff9b1e2fbe467d23276236aa52fb37eb73fd340da91
for dv_infpl - show
Inserted entry with lex_id: 51 into lexicon
Linking lex_id 51 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 51 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 7cf576b300bbb026e4a7b5b4a8d792dc0f6e834354a4efe471977f2d71517a9a
for dv_pp - shown
Inserted entry with lex_id: 52 into lexicon
Linking lex_id 52 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 52 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 42f5f444d7d957ad84c5a633757577db59591b988579f53bd1f350baddaf9f03
for dv_finsg - shows
Entry already exists for hash:
42f5f444d7d957ad84c5a633757577db59591b988579f53bd1f350baddaf9f03
Generated hash: d14f9f05e5a0b45180e21ff9b1e2fbe467d23276236aa52fb37eb73fd340da91
for dv_infpl - show
Entry already exists for hash:
d14f9f05e5a0b45180e21ff9b1e2fbe467d23276236aa52fb37eb73fd340da91
Generated hash: 7cf576b300bbb026e4a7b5b4a8d792dc0f6e834354a4efe471977f2d71517a9a
for dv_pp - shown
Entry already exists for hash:
7cf576b300bbb026e4a7b5b4a8d792dc0f6e834354a4efe471977f2d71517a9a
Generated hash: 4b9730c9ceecf1d5f810a4826903bb059c3e92371075ace397c16ed7d738418d
for dv_finsg - forgives
```

```
Inserted entry with lex_id: 53 into lexicon
Linking lex_id 53 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 53 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 9b948431ccbbc7fe72acc6521239aa45db87345868a085fef14be2952cf73343
for dv_infpl - forgive
Inserted entry with lex_id: 54 into lexicon
Linking lex_id 54 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 54 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 87c13947298aa7164ca11f17ffb7cf6e0725519849e9cd3b369c99e69db9f417
for dv_pp - forgiven
Inserted entry with lex_id: 55 into lexicon
Linking lex_id 55 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 55 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 1b1a64cc34090572aa43370164020fb77fe5e26ea27e6841fb5da311c70db49e
for dv_finsg - succeeds
Inserted entry with lex_id: 56 into lexicon
Linking lex_id 56 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 56 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 94cd50411086a27d99e36cea494fca14e31bfcdde6608e91b7d21b8abbfcba24
for dv_infpl - succeed
Inserted entry with lex_id: 57 into lexicon
Linking lex_id 57 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 57 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 8ee745975fad537905042b710e2f602f6c6bbe6c72f123b3596ce0b962f2b23f
for dv_pp - succeeded
Inserted entry with lex_id: 58 into lexicon
Linking lex_id 58 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 58 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Generated hash: 49416a90f7e5b5ab216b186657fa6a7e5219aaa07c0d4fc0dfa1373b2f2663b9
for prep - in
Inserted entry with lex_id: 59 into lexicon
Linking lex_id 59 with stix_object_id x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Successfully linked lex_id 59 with stix_uuid x-stixd-clex--
a18cb71a-24bd-4f8a-9d57-0a7baf7181a0
Import successful.
```

```
New entries imported:    59
Existing entries linked: 3
================================
Total entries processed: 62
```

1.7.4 Show Database State After Code Execution

After running the Clex Importer tool, the tables below should have the expected number of rows.

- Rows in 'stix_objects' table: 1
- Rows in 'lexicon' table: 59
- Rows in 'obj_lex_jt' table: 59

Let's verify the state of the database by running a query to count entries in each table.

[ ]:
```
# Fetch table counts
lexicon_count = %sql SELECT COUNT(*) FROM lexicon;
stix_objects_count = %sql SELECT COUNT(*) FROM stix_objects;
obj_lex_jt_count = %sql SELECT COUNT(*) FROM obj_lex_jt;

# Display number of rows in each table
print(f"\nRows in 'stix_objects' table:  {stix_objects_count[0][0]}\n"
      f"Rows in 'lexicon' table:      {lexicon_count[0][0]}\n"
      f"Rows in 'obj_lex_jt' table:   {obj_lex_jt_count[0][0]}")
```

```
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.

Rows in 'stix_objects' table:  1
Rows in 'lexicon' table:      59
Rows in 'obj_lex_jt' table:   59
```

We will also display a sample of the first five entries in each table to demonstrate the successful importation of Clex entries.

[ ]:
```
# Display the first 5 rows of the lexicon table
%sql SELECT * FROM stixd_corpus.lexicon LIMIT 5;
```

```
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
5 rows affected.
```

[ ]:
```
[(1, 'adv', 'fast', 'fast', 'NULL',
'67e9b1c5cbd53045919deda792be49b18b41a09b3bd328f9cc406bb27d951f62', None, None,
None),
 (2, 'adv_comp', 'faster', 'fast', 'NULL',
'38a31bf0527ff6fd23c6be74bfba58c46dbad709ce90b6d09b9a26f103a326b5', None, None,
None),
 (3, 'adv_sup', 'fastest', 'fast', 'NULL',
'55fee0f355e343b2c6a4d63b72a8ea8bcaa1a71698ada04e01533a8dc98fb4ee', None, None,
None),
 (4, 'adv', 'quickly', 'quickly', 'NULL',
'b0a248290b9aa18bfbbbfd5367dc0cc0dc82a9e90dd83b88cce59361b8d67e8a', None, None,
None),
 (5, 'adj_itr', 'large', 'large', 'NULL',
'bbe9bafa7a2a6e250fdf482a7c46217d7c63ccee917b3ae48324b61659c7e32d', None, None,
None)]
```

```
[ ]: # Display the first 5 rows of the stix_objects table
     %sql SELECT * FROM stixd_corpus.stix_objects LIMIT 5;
```

```
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
1 rows affected.
```

```
[ ]: [('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 'x-stixd-clex', 'user',
     'ACE Common Lexicon Import', '2.1', datetime.datetime(2024, 8, 17, 14, 8, 55),
     datetime.datetime(2024, 8, 17, 14, 8, 55), 0, '["lexicon"]', 100, 'en', '[]',
     '[]', '[]', '[]', None, None, '[]', None)]
```

```
[ ]: # Display the first 5 rows of the obj_lex_jt junction table
     %sql SELECT * FROM stixd_corpus.obj_lex_jt LIMIT 5;
```

```
 * mysql+mysqlconnector://your_username:***@localhost:3306/stixd_corpus
5 rows affected.
```

```
[ ]: [('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 1),
      ('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 2),
      ('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 3),
      ('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 4),
      ('x-stixd-clex--a18cb71a-24bd-4f8a-9d57-0a7baf7181a0', 5)]
```

1.7.5 Executing the Clex Importer via Web Form

While the Flask API cannot be run directly within this notebook, you can run it locally by following these steps:

1. Navigate to the directory containing the `api.py` file.
2. Activate the virtual environment:
    - On Windows: `.venv\Scripts\activate`
    - On macOS/Linux: `source .venv/bin/activate`
3. Install dependencies, if necessary:
    - `pip install -r requirements.txt`
4. Run the Flask API:
    - `python api.py`
5. Access the web form at `http://localhost:5000/`
6. When finished, stop the Flask API by pressing `Ctrl+C` in the terminal.

1.8 Conclusion

In this demonstration, we explored the Clex Importer tool, which populates the `lexicon` table in the STIX-D Corpus Database with entries from the ACE Common Lexicon (Clex). By importing lexical entries from the Clex file, the tool enables precise language processing for applications requiring unambiguous interpretation by both humans and machines. The project design leverages object-oriented programming principles to create a modular, extensible, and maintainable system, with key modules like `ClexImporter` and `MySQLRepository` facilitating database interactions and lexicon importation. The testing strategy includes unit, integration, and end-to-end tests to ensure the reliability and correctness of the codebase. By executing the Clex Importer tool via the command line interface and web interface, users can seamlessly import Clex entries into the database, supporting ACE-based natural language processing tasks.