Enrique Juan

MongoDB Assignment

Aircraft wildlife strikes

Data Loading Script:

Introduction:

The data loading script serves the purpose of ingesting data from the CSV file into a MongoDB database. It uses Pandas for efficient handling of the CSV's data and PyMongo for interaction with MongoDB.

Data Loading and Transformation:

The script begins by loading the 'database.csv' file into a Pandas DataFrame named `new_df`. Then it prints the column names to ensure the correct dataset is loaded. Date columns are all converted to a date-time format, and special consideration is given to handle mixed types in the 'Record ID' column.

MongoDB Connection:

To establish a connection with MongoDB, the script uses the PyMongo library, providing a specified URI and necessary authentication details.

Database and Collection Creation:

The script creates a MongoDB database named "AirIncidents" and a collection named "Incident" using the MongoClient object. Any existing data in the "Incident" collection is cleared at the beginning to ensure a clean slate.

Document Creation and Insertion:

MongoDB documents are created for each incident in the DataFrame. The script iterates through each row, extracting relevant information, and constructing a document. It checks for the existence of the 'Record ID' column before proceeding with the insertion into the MongoDB collection.

Validation Steps:

Check for 'Record ID' Column:

The script validates the presence of the 'Record ID' column before inserting data into the database. If the column is not found, a message is printed, and the script stops execution.

Data Type Conversion:

Certain columns are explicitly converted to string type to ensure uniform data types in MongoDB documents.

Date Parsing:

Date columns are converted to date-time format using a specific format, and parsing errors are handled with the 'coerce' option, replacing invalid dates with NaN.

Empty Data:

If in a row no data is found and it is in a non-essential column to the creation of the document (any except the Record ID column), the value “nan” is given to fill up the missing data.

MongoDB Insertion:

Each document is inserted into the MongoDB collection using the `insert_one` method.

Closing MongoDB Connection:

After successfully inserting all documents, the script closes the MongoDB connection using the `close` method to free up resources.

Error Handling:

- If the 'Record ID' column is not found, a clear message is printed, alerting the user to check the CSV file and script configuration before retrying.
- Date parsing errors are mitigated with the 'coerce' option, ensuring that invalid dates do not compromise the data loading process.
- If no data is found in a position where it is not essential, it is filled with the string “nan” so to keep the program going and not to interfere with future operations.

Conclusion:
The data loading script, with its comprehensive approach to handling data, ensures the integrity of information being transferred from the CSV file to the MongoDB database.

MongoDB Query Program:

Introduction:
This program facilitates the querying and modification of MongoDB documents within the "Incident" collection. It is designed to be useful, accepting command-line arguments for various query and modification commands.

Query Functions:

Query All Documents:
Lists all document IDs in the "Incident" collection, providing an overview of the available records.
Query Documents by species hit:
Lists document IDs matching a specified bird species name. This focuses the impact on wildlife.
Query with Projection:

Allows users to select specific fields for all documents.

Query and Sort:

Lists document IDs sorted by a specified field and order, facilitating ordered data exploration.

Aggregation Query:

Performs an aggregation query using match criteria.

Specific Query:

Shows all data in a document matching the record ID given, offering an in-depth study of the document.

Modification Functions:

Add Document:

Adds a new document to the database, enabling the addition of new incidents into the dataset.

Update Document:

Updates a document in the database based on the record ID for corrections or additions to existing records.

Delete Document:

Deletes a document from the database based on the record ID.

Command-Line Arguments:

The program uses argparse library for command-line parsing and modifying the database. Queries include all, specific, species, projection, sorted, and aggregation, and modifications include add, update, and delete.

Users can refer to the —help command to view the syntax for executing specific queries or modifications.