This report presents the methodology employed in constructing a JSON database from multiple datasets, followed by converting this database into CSV format. The process has been broken down into steps to ensure comprehension and replication. The dataset utilized in this study was sourced from the FAO and the World Bank and was initially compiled in CSV format before being converted to JSON and back to CSV. Although converting datasets from CSV to JSON and then back to CSV may appear circuitous, it serves several purposes that enhance data handling, analysis, and storage. The following are the primary justifications for this approach:

**1. Structured Data Aggregation and Normalization**

**Centralized Data Repository:**

* **Unification**: Multiple datasets with varying structures and fields are unified under a single schema in the JSON format. This creates a centralized data repository, simplifying access and management.
* **Normalization**: The JSON schema allows for the normalization of data fields across datasets, ensuring consistency. This is particularly valuable when dealing with datasets from diverse sources or with different formats.

**2. Flexibility and Hierarchical Data Representation**

**JSON's Hierarchical Nature:**

* **Flexibility**: JSON format is highly flexible, representing complex data structures beyond flat tables. This is useful for datasets with nested or hierarchical data.
* **Handling Complexity**: JSON can represent these relationships more naturally than CSV for datasets with multi-level hierarchical data.

**3. Data Integrity and Transformation**

**Ensuring Accuracy:**

* **Data Integrity Checks**: Transforming data into JSON and then back to CSV includes steps to ensure data integrity, such as format consistency and value accuracy.
* **Transformations and Cleaning**: The intermediate JSON format allows for efficient data transformations and cleaning processes, which can be more cumbersome directly in CSV.

**4. Scalability and Future-Proofing**

**Adapting to Growing Data Needs:**

* **Scalability**: JSON databases are more scalable for larger datasets or when combining multiple data sources.
* **Future Expansion**: If data needs evolve (e.g., adding more datasets or changing structures), the JSON schema can be easily adjusted, whereas CSV is more rigid.

**5. Intermediate Step for Advanced Data Processing**

**Enabling Complex Operations:**

* **Data Analysis and Processing**: The conversion to JSON facilitates complex data analysis and processing, leveraging advanced programming and database management techniques.
* **Preparation for Advanced Applications**: This approach prepares the data for more advanced applications, such as integration with web APIs, feeding into machine learning models, or use in NoSQL databases.

**Step 1: Designing the JSON Schema**

**Objective:**

To create a versatile and comprehensive schema that can accommodate various datasets with differing structures and data types.

**Procedure:**

1. **Analyze Dataset Structures**: Reviewed each dataset's structure and key details, including variables and data types.
2. **Identify Common Fields**: Determined fields common across all datasets (e.g., 'Area Code,' 'Year,' 'Value').
3. **Define Specific Fields**: Identified dataset-specific fields to capture unique aspects (e.g., 'Domain Code,' 'Element Code' for agricultural data).
4. **Develop a Flexible Schema**: Created a schema that includes common and specific fields, ensuring flexibility to handle diverse data.

**Step 2: Building the JSON Database**

**Objective:**

To accurately and efficiently transfer data from individual datasets into the structured JSON database.

**Procedure:**

1. **Initialize JSON Database**: Set up an empty JSON structure with collections corresponding to each dataset.
2. **Data Transformation**:
   * For each dataset, read the data (typically in CSV format).
   * Transform each row into a dictionary, aligning with the JSON schema.
3. **Data Loading**:
   * Append these dictionaries to the respective collection in the JSON database.
   * Ensure that each entry in the collection represents a row from the dataset.
4. **Data Integrity Check**: Conducted audits to ensure completeness and accuracy of the data transfer.

**Step 3: Filtering the JSON Database**

**Objective:**

To filter the database for a specific time range (2016-2021) while maintaining data integrity.

**Procedure:**

1. **Define Filtering Criteria**: Established the year range (2016-2021) as the filter criterion.
2. **Apply Filters**:
   * Iterated over each record in the collections.
   * Included records where the year (or start year in case of a range) falls within the specified range.

**Step 4: Converting JSON to CSV**

**Objective:**

To convert the structured JSON database into individual flat-file CSV formats for each collection.

**Procedure:**

1. **Prepare CSV Files**: Set up separate CSV files corresponding to each collection in the JSON database.
2. **Data Conversion**:
   * For each collection, convert the list of dictionaries into a pandas DataFrame.
   * Each DataFrame was saved as a CSV file, preserving the structure and data types.

The methodology outlined ensures a structured approach to handling diverse datasets, accommodating them in a unified JSON schema and allowing for efficient conversion to CSV. Following these steps, the procedure can be replicated for data aggregation and format conversion tasks.