Outer

Constructor: Map : <id, dataset object>

↓

check key attributes

---

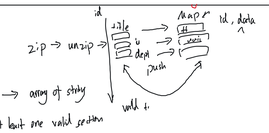
export default class Section {

key attribute.

}

---

call helper



Helper

variable id

Data → { 1 section ✓ → array of string
n section ✓ at least one valid section

("dataset ID", {"title": "value" ..... })

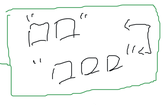
const MapXx : Map<string, { valid field : valid field }> = new Map() this.validDataset = new Map<string, any[]>();

section [ ]

↳ { title : "string"
dept : string

* define : valid list [key attributes]

addDatasets (id, content, kind)



if (id == null || id.includes("_") || id == "" || id.trim().length == 0)

if id already exists if (i in this.Map) → reject

if (kind == section - ....

if check content

JSZip.loadAsync(content)

Private loadZip ( )

→ if not zip, throw error

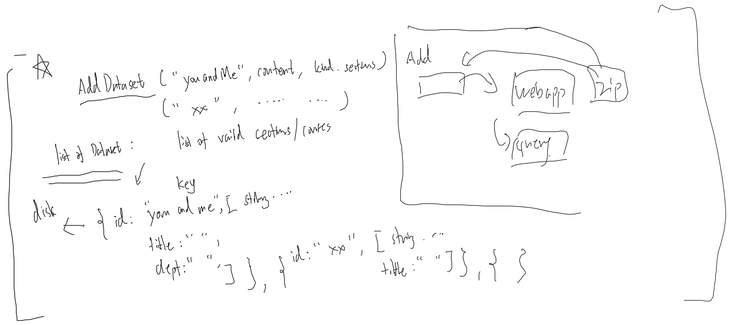
→ if zip, find course { not, throw
found, loop Json: 1. parse

3. compare valid list

→ check numRows

→ validate / modify : id + "_dept" ... when to add? Ask TA! at checkin

---



---

convert to data model with map

1. DatasetProcessor Class JSZip

Responsibilities:

✓ Load Datasets:
Utilize file system operations to read and load datasets from the provided zip files.
Handle errors and ensure graceful handling of file-related issues.

✓ Validate Datasets:
Implement a DatasetValidator or integrate validation logic within the DatasetProcessor to check the integrity and correctness of the loaded datasets.
Verify that the datasets adhere to the specified format, including checking for required fields, data types, etc.

✓ Convert to Data Model:
~~Define a data model class~~ (e.g., Section) that represents the structure of the dataset.
Convert the raw dataset into instances of this data model for ease of processing in the Query Engine.

? Process ~~and S~~tore Datasets:
Apply ~~any nec~~essary pre-processing steps to enhance data quality or structure.
Store the processed datasets in a suitable data structure that allows efficient querying.

Error Handling:
Implement error handling mechanisms to gracefully handle issues such as invalid datasets, missing files, or unexpected data structures.