INFO 7255 Advanced Big Data Indexing Techniques

Getting Started with JSON Schema

Introduction

- JSON Schema is a specification for defining the structure of JSON data. It was written under IETF draft which expired in 2011.

- https://www.json.org/json-en.html
- https://json-schema.org/specification.html

Type Information to Json document

- Describes your existing data format.
- Clear, human- and machine-readable documentation.
- Complete structural validation, useful for automated testing.
- Complete structural validation, validating client-submitted data.
- Let's the check various important keywords that can be used in this schema –

required - This keeps a list of required properties.

minimum - This is the constraint to be put on the value and represents minimum acceptable value.

maximum - This is the constraint to be put on the value and represents maximum acceptable value.

maxLength - The length of a string instance is defined as the maximum number of its characters.

minLength - The length of a string instance is defined as the minimum number of its characters.

pattern - A string instance is considered valid if the regular expression matches the instance successfully.

- Let's pretend we're interacting with a JSON based product catalog. This catalog has a product which has:
  - An identifier: `objectId`
  - An object name: `objectName`
  - A selling cost for the consumer: `price`
  - An optional set of tags: `tags`.
- For example:

```
{
  "objectId": 1,
  "objectName": "A green door",
  "price": 12.50,
  "tags": [ "home", "green" ]
}
```

- While generally straightforward, the example leaves some open questions. Here are just a few of them:
  - What is `objectId`?
  - Is `objectName` required?

Starting the schema

- To start a schema definition, let's begin with a basic JSON schema.
- We start with four properties called **keywords** which are expressed as JSON keys.
    1. The `$schema` keyword states that this schema is written according to a specific draft of the standard and used for a variety of reasons, primarily version control.
    2. The `$id` keyword defines a URI for the schema, and the base URI that other URI references within the schema are resolved against.
    3. The `title` and `description` annotation keywords are descriptive only. They do not add constraints to the data being validated. The intent of the schema is stated with these two keywords.
    4. The `type` validation keyword defines the first constraint on our JSON data and in this case it has to be a JSON Object.

```
{
"$schema": "http://json-schema.org/draft-07/schema#",
"$id": "http://example.com/product.schema.json",
"title": "Product",
"description": "A product in the catalog",
"type": "object"
```

Defining the properties

- objectId is a numeric value that uniquely identifies a product. Since this is a canonical identifier for a product, it doesn't make sense to have an object without one, therefore it is "required"
- *objectName* is a string value that describes an object. This is also for the same above reasons required

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://example.com/product.schema.json",
  "title": "Object",
  "description": "An object in the catalog",
  "type": "object",
  "properties": {
    "objectId" : {
      "description" : " The unique identifier of object ",
      "type" : "integer",
    },
    "objectName" : {
     "description" : " The name of object ",
     "type" : "string"
    }
  },
 "required" : ["objectId" , "objectName"]
}
```

- Similarly, Price key and as the price must be greater than zero, "exclusiveMinimum" validation keyword should be used
- Now, If there are tags there must be at least one tag, unique, no duplication within a single object.
- All tags must be text, aren't required to be present.

Online JSON Schema Validator Link:

https://www.jsonschemavalidator.net/

For these requirements we would design the schema as follows:

```
{ "$schema": "http://json-schema.org/draft-07/schema#",
"$id": "http://example.com/product.schema.json",
"title": "Objectt",
"description": "An object from the catalog",
"type": "object",
"properties": {
  "objectId": {
                "description": "The unique identifier for an objectt",
                "type": "integer"
                },
  "objecttName": {
                "description": "Name of the object",
                "type": "string"
                },
  "price": {
                "description": "The price of a particular object",
                "type": "number",
                "exclusiveMinimum": 0
                },
  "tags": {
                "description": "Tags – uses the array property of schema",
                "type": "array",
                "items": {
                    "type": "string"
                },
                "minItems": 1,
                "uniqueItems": true
                } },
                 "required": [ "objectId", "objectName", "price" ]
}
```

Data

```
{
   "objectId": 1,
   "objectName": "A green door",
   "price": 12.50,
   "tags": [ "home", "green" ]
}
```

Easy-To-Learn-Image-for JSON Schema