MongoDB

Design pattern exercises (2)

DANIELE APILETTI

POLITECNICO DI TORINO

Exercise 4 – Parcel delivery (1)

- Design a MongoDB database to manage the parcel delivery according to the following requirements.
- Customers of the parcel delivery service are citizens identified by their social security number. They can be senders or recipients of delivered parcels. They are characterized by their name, surname, email address, a telephone number, and by different addresses, one for each type (e.g., one billing address, one home address, one work address, etc.). Each address consists of street name, street number, postal code, city, province, and country.
- Parcels are characterized by a unique barcode and their physical dimensions (specifically: width, height, depth, and weight). All widths, heights, and depths are always expressed in meters. All weights are always expressed in kilograms.
- The recipient and the sender information required to deliver each parcel must be always available when accessing the data of a parcel. Recipient and sender information required to deliver a parcel consists of full name, street name, street number, postal code, city, province, and country.

Exercise 4 – Parcel delivery (2)

- The parcel warehouse is divided into different areas. Each **area** is identified by a unique code, e.g., 'area_51' and consists of different lines. Each **line** is identified by unique code, e.g., 'line_12', and hosts several racks. Each **rack** is identified by unique code, e.g., 'rack_33', and is made up of shelves. Each parcel is placed on a specific **shelf** of the warehouse, identified by a unique code, e.g., 'shelf_99'. The database is required to track the location of each parcel within the warehouse.
- Given a parcel, the database must be designed to efficiently provide its full location, from the shelf, up to the area, through rack and line.
- Given a customer, the database must be designed to efficiently provide all her/his parcels as a sender, and all his/her parcels as a recipient.
  - Indicate for each collection in the database the document structure and the strategies used for modelling

Exercise 4 – Parcel delivery

```
Parcel
{_id: <string>, // barcode
 dimensions: { // also 1st-level attribute is fine
     width: <number>,
     height: <number>,
     depth: <number>,
     weight: <number>
 },
 recipient: {
     _id: <string>, // e.g., SSN, id of the customer
     street: <string>,
     number: <string>,
     zip_code: <string>,
     city: <string>,
     province: <string>
 },
 sender:{
     _id: <string>,// e.g., SSN, _id of the customer
     street: <string>,
     number: <string>,
     zip_code: <string>,
     city: <string>,
     province: <string>
 },
 father_pos: <string>, // code of area/lane/rack/shelf
 locations: [
     <string> // list of codes of area/lane/rack/shelf
 ]
}
```

Extended reference pattern for recipient and sender address information. The recipient and sender _ids are required to look up all parcels of a given customer.

Tree pattern for the position. The full list of tree-pattern ancestors is required. The parent ancestor of the tree-pattern is optional.

(No collection for the areas, since no data are tracked except their code)

Exercise 4 – Parcel delivery

```
Customers
{_id: <string>, // fiscal code
 name: <string>,
 surname: <string>,
 email: <string>,
 tel: <string>,
 addresses:{
     'home': {
          street_name: <string>,
          street_num: <string>,
          postal_code: <int>,
          city: <string>,
          province: <string>,
          country: <string>
     },
     'billing':{
          street_name: <string>,
          street_num: <string>,
          postal_code: <int>,
          city: <string>,
          province: <string>,
          country: <string>
     },
     'work': {...}
 }
}
```

Attribute pattern (optional) for the addresses attribute.

Exercise 5 – «Museum exhibitions»

- We want to design the database to manage museum exhibitions according to the following requirements.
- Museums are characterized by their name, address, a telephone number, and a website (if available). The address consists of geographical coordinates, street name and number, postal code, and city.
- The items exhibited in the museums are identified by a progressive number and characterized by a title, a description and the list of author names. The items are categorized as either archaeological finds, or paintings, or sculptures.
- The database must record all the main features of each item, such as its dimensions (i.e., width, height, weight, etc.). Each feature has at least a name and a value, and possibly a unit of measure. For instance, the main material is a feature of an archaeological find, the geometrical sizes are features of a painting. For each item, the museum to which it belongs must be recorded, with the museum name frequently accessed together with the item itself.

Exercise 5 – «Museum exhibitions»

- Several exhibitions are hosted in each museum. The exhibition is characterized by a title, a description, the list of curator names. You must record all the items associated with each exhibition; they can be in the order of hundreds. An item can be part of different exhibitions. Moreover, each exhibition can be hosted by several museums in different periods. You must record the start and end dates of each exhibition in each museum.
- Given an item, the database must be designed to efficiently provide the name of the museum that owns it.
- Given an exhibition, the database must be designed to efficiently provide the name of the museum and the geographical coordinates where it has been hosted.
- Furthermore, given an exhibition, the list of items included in the exhibition and their number must be efficiently returned.
    - Indicate for each collection in the database the document structure and the strategies used for modelling.

Exercise 5 – «Museum exhibitions»

```
Museums
{_id: ObjectId(),
 name: <string>,
 address: {
     street: <string>,
     number: <string|number>,
     postal_code: <number>,
     city: <string>,
     province: <string>,
     geo_ref: {
          type: <string>,
          coordinates: [ <number> ]
     }
 },
 phone: <string>,
 website: <string> // optional
}
```

Polymorphic pattern to track the museum information in the museum collection.

```
Items
{ _id: <number>,
 title: <string>,
 description: <string>,
 authors: [ <string> ],
 category: <string>,
 features: [ {k: <string>, v: <string>, u: <string>} ],
 museum: {
     _id: ObjectId(),
     name: <string>
 }
}
```

Attribute pattern (with polymorphic pattern) to track the features of each item.

Extended reference pattern to track the museum information associated with each item.

Exercise 5 – «Museum exhibitions»

```
Exhibitions
{_id: ObjectId(),
 title: <string>,
 description:<string>,
 curators: [ <string> ],
 events: [
    {start: <date>,
     end: <date>,
     museum:{
        _id: itemId(),
        name: <string>,
        geo_ref: {
            type: <string>,
            coordinates: [ <number> ]
        }
    }
 ],
 items: [<number> ] // _id of items }
}
```

Bucket pattern with **extended** reference pattern to track when an exhibition is hosted in a museum.

Acknowledgment

Bibliography

For further information on the content of these slides, please refer to the book

“*Design with MongoDB*”

Best Models for Applications

by Alessandro Fiori

https://flowygo.com/en/projects/design-with-mongodb/