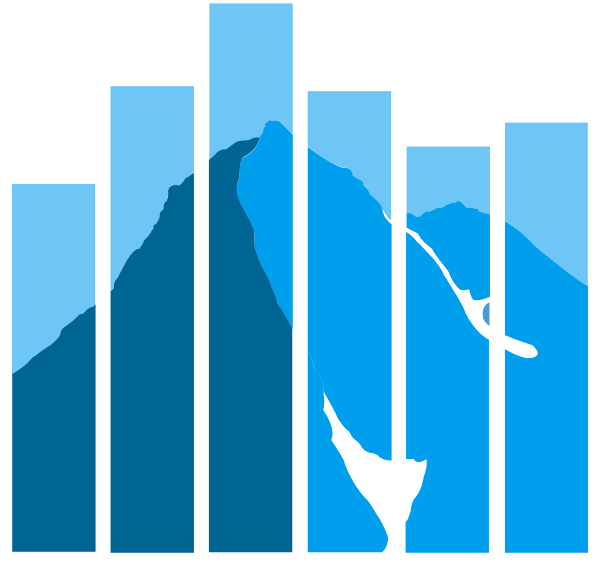


THE LOANPRO API

WORKING WITH ELASTICSEARCH (PART 2)

loanprosoftware.simnang.com

API SAMPLES

Fork Customer Search — https://plnkr.co/edit/xmemSm

Fork Loan Search — https://plnkr.co/edit/1MrwiP

⚠ CAUTIONARY NOTE ⚠

The provided API samples store API credentials in the browser

Use the API samples to explore the API and not as your integration

WHAT WAS COVERED IN PART 1?

- How LoanPro uses Elasticsearch
- Basics of Elasticsearch query objects
- Basics of aggregation in Elasticsearch

WHAT WAS NOT COVERED IN PART 1?

- What fields are available for searching
- How to search nested entities
- Common pitfalls and how to avoid them

AVAILABLE FIELDS

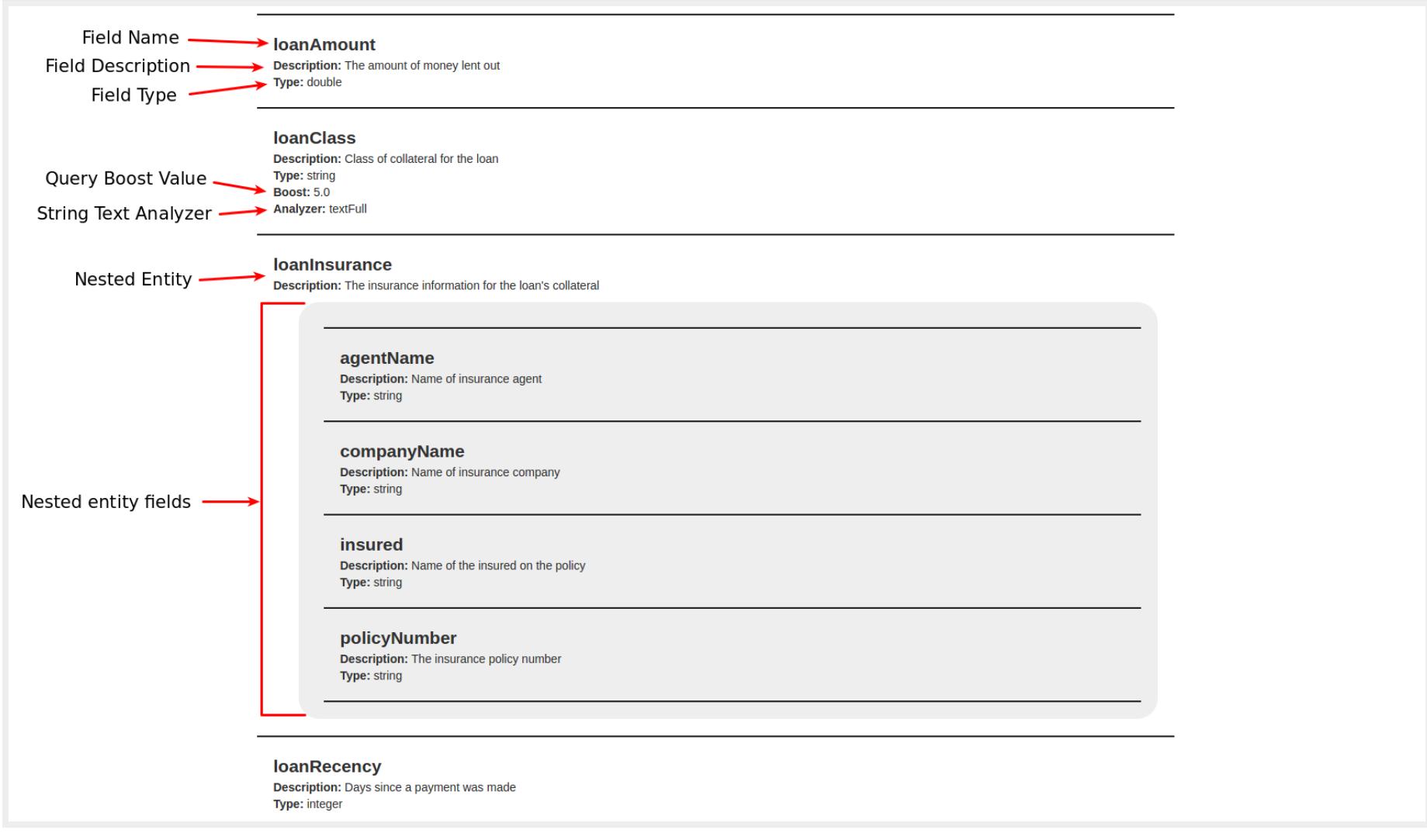
LoanPro provides a database index of fields that are available

<<<>>>

UNDERSTANDING THE DOCUMENTATION

- Each section starts with the field name
- Description of what the field is is below
- Type of field is below the Description
  - Can be integer, double, string, object, date, or nested
  - Nested objects have their fields indented and with a grey background
- Some optional parameters will appear when available
  - Boost - the amount of ranking weight that Elasticsearch gives the field
  - Analyzer - the text analyzing engine used by Elasticsearch
  - These two options aren't used in 99% of searches

SAMPLE DOCUMENTATION



NESTED OBJECTS

Nested objects have two parts:

- The name of the nested objects
- The fields of the nested object

To search a field of a nested object, your field name will be the name of the nested object, ".", name of the field

- eg. "loanInsurance.agentName" - searches by name of insurance agent

NESTED OBJECT EXAMPLE

```
"query_string": {
  "query": "*bob*",
  "default_operator": "OR",
  "fields": [
    "collateral.vin",
    "collateral.color",
    "collateral.gpsCode",
    "collateral.licensePlate",

    "primaryCustomerAddress.city",
    "primaryCustomerAddress.zipcode",
    "primaryCustomerAddress.state",

    "secondaryCustomerAddress.city",
    "secondaryCustomerAddress.zipcode",
    "secondaryCustomerAddress.state",

    "loanInsurance.companyName",
    "loanInsurance.insured",
    "loanInsurance.policyNumber",
    "loanInsurance.agentName"
  ]
```

Checks collateral info, primary customer address, secondary customer address, and loan insurance for the phrase "bob"

COMMON PITFALLS

Common pitfalls include:

- Mixing up "must" and "should"
- Trying to access a nested property without specifying the nested object
- Forgetting to wrap "must" or "should" in a "bool"
- Forgetting to wrap "match" or "query_string" in an object inside of "must" or "should"
- Not sending over the correct search
- Sending the query object in the wrong field

AVOIDING PITFALLS

- Write out the desired query in familiar syntax first, then translate it
- Use a JSON editor to make sure you understand how your query is formed
  - Make sure child objects are in fact child objects
  - Make sure child objects are children of the correct parent object
- Read the documentation for the search your performing
- Test, test, test!

QUESTIONS?

Feel free to ask in our Forums

Or, email us at support@simnang.com