



SnipMan

Organize your Code Snippets

Group 42

Péter Ferenc Gyarmati 11913346

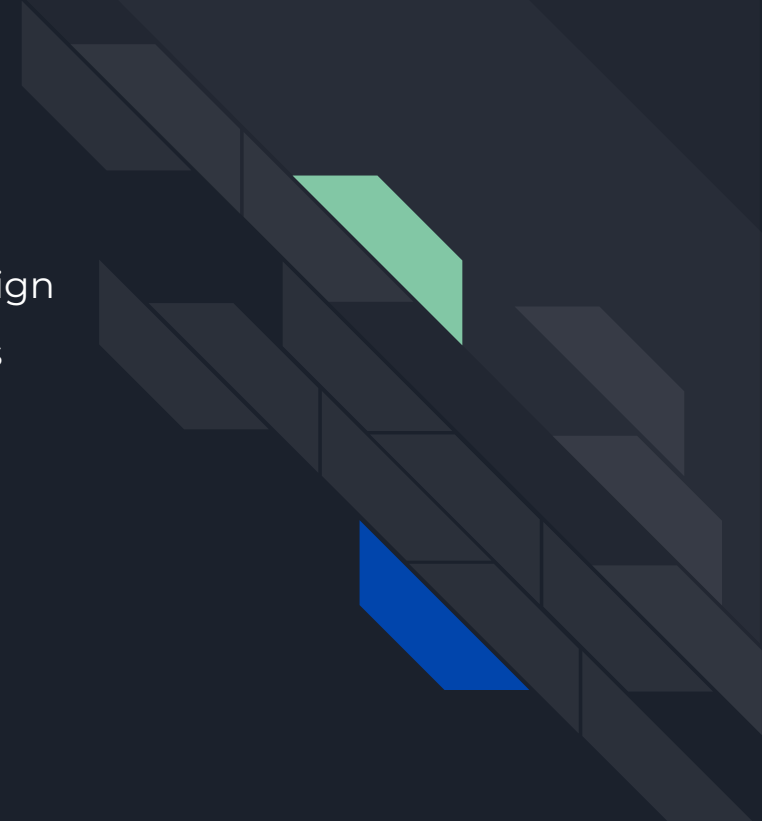
Simon Eckerstorfer 11911424

Overview

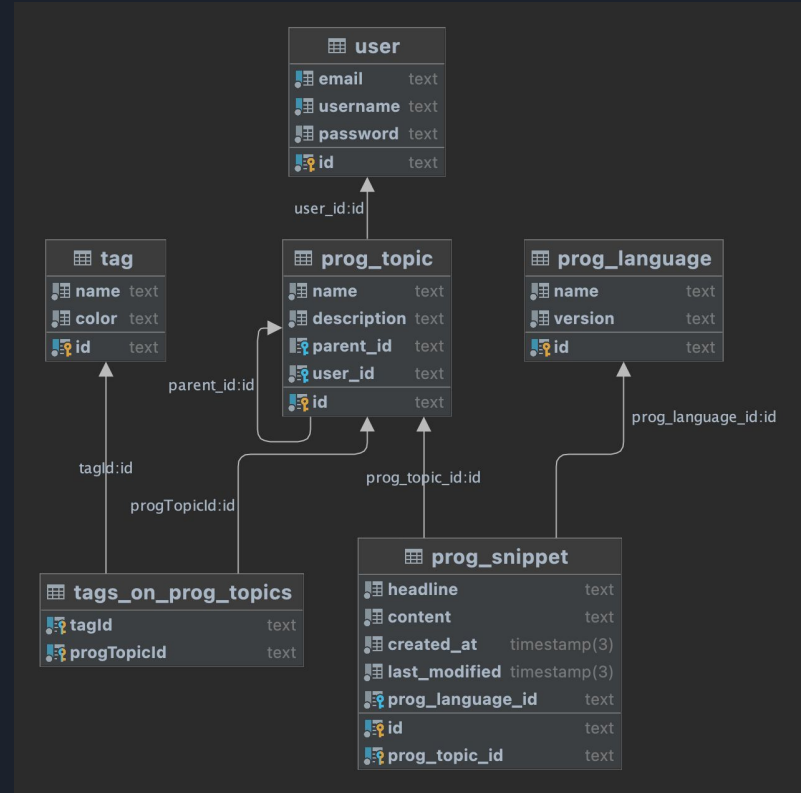
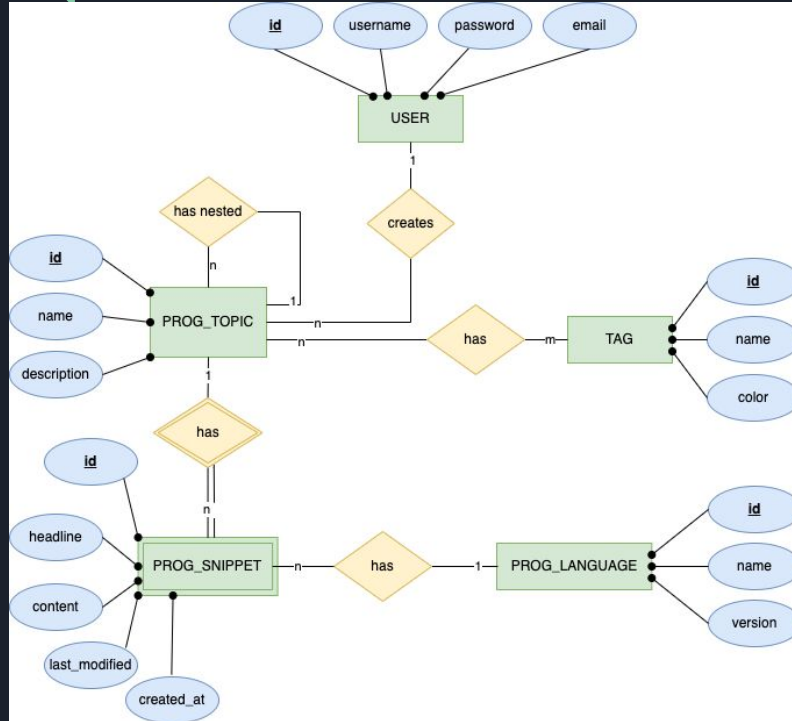
Comparison of RDBMS model and NoSQL Design

Performance Implications of NoSql for Reports

Query Comparison



Comparison of RDBMS model and NoSQL Design



Comparison of RDBMS model and NoSQL Design

- One collection for each data model

<p>prog_language</p> <p>Storage size: 36.86 kB</p> <p>Documents: 25</p> <p>Avg. document size: 54.00 B</p> <p>Indexes: 1</p> <p>Total index size: 36.86 kB</p>	<p>prog_snippet</p> <p>Storage size: 53.25 kB</p> <p>Documents: 93</p> <p>Avg. document size: 542.00 B</p> <p>Indexes: 1</p> <p>Total index size: 36.86 kB</p>	<p>prog_topic</p> <p>Storage size: 36.86 kB</p> <p>Documents: 30</p> <p>Avg. document size: 398.00 B</p> <p>Indexes: 1</p> <p>Total index size: 36.86 kB</p>
<p>tag</p> <p>Storage size: 36.86 kB</p> <p>Documents: 8</p> <p>Avg. document size: 61.00 B</p> <p>Indexes: 1</p> <p>Total index size: 36.86 kB</p>	<p>user</p> <p>Storage size: 36.86 kB</p> <p>Documents: 3</p> <p>Avg. document size: 296.00 B</p> <p>Indexes: 3</p> <p>Total index size: 110.59 kB</p>	



Comparison of RDBMS model and NoSQL Design

- **User** collection
- Topics big and nested -> only IDs

```
_id: ObjectId('62ab09fb89987f389fc218c5')
email: "Laverne_Carroll71@hotmail.com"
username: "Laverne_Carroll71"
password: "Kp4qWYmHlf0zlkG"
✓ prog_topic_ids: Array
  0: ObjectId('62ab09fb89987f389fc218c8')
  1: ObjectId('62ab09fb89987f389fc218c9')
```

- **Tag** collection for efficient queries
- Tiny documents, mostly read with topics, rarely written
 - also embedded in topics (see next slide)

```
_id: ObjectId('62ab09fc89987f389fc218e6')
name: "svelte-components"
color: "#18771a"
```



Comparison of RDBMS model and NoSQL Design

- **Topics** collection
- Snippets not limited in size, may get very big -> only IDs
- Nesting depth of topics not limited -> only parent ID
 - No parent -> root topic
- Tags needed on every topic read -> embedded

```
_id: ObjectId('62ab09fb89987f389fc218c9')
name: "transmit"
description: "Use the digital SQL array, then you
user_id: ObjectId('62ab09fb89987f389fc218c5')
✓ tags: Array
  ✓ 0: Object
    name: "timeline"
    color: "#522a07"
  > 1: Object
  > 2: Object
parent_id: ObjectId('62ab09fc89987f389fc218e0')
✓ prog_snippet_ids: Array
  0: ObjectId('62ab09fc89987f389fc218f1')
```

Comparison of RDBMS model and NoSQL Design

- **Snippet collection**
- Weak entity, but may get big -> related topic ID saved
- Programming Languages are tiny + needed on most reads -> embedded
- User Email needed for report -> included in document

```
{
  _id: ObjectId('62ab09fc89987f389fc218f1')
  headline: "Select database table and populate"
  content: "$result = $mysqli->query('SELECT * FROM students');
           while ($row = $res..."
  created_at: 2022-06-16T10:44:47.334+00:00
  last_modified: 2022-06-16T10:44:47.335+00:00
  user_email: "Laverne_Carroll71@hotmail.com"
  prog_topic_id: ObjectId('62ab09fb89987f389fc218c9')
  prog_language: Object
    name: "PHP"
    version: "8.0"
```

- **Programming Languages similar to Tags** - tiny documents
 - Own collection for efficient query
 - Embedded in snippet

```
{
  _id: ObjectId('62ab09fb89987f389fc218ac')
  name: "C++"
  version: "20"
```



Comparison of RDBMS model and NoSQL Design

- More flexible data models using NoSQL
- The NoSQL design made it possible to customize the stored documents' structure to be ideal for our exact use cases
- Data that is accessed together is stored together (mostly)



Performance Implications of NoSQL for Reports

Active Users Report (Péter)

- Programming Language and User Email address embedded
- Projection and filtering before data joining
- TTL Index for *prog_snippet.created_at*

Dominant Languages Report (Simon)

- Programming Languages are embedded -> no join necessary
- Lookup still needed for snippets (Snippets not embedded on Topics)
- Projecting and matching
 - In particular before lookup

Query Comparison - **Active Users**

Joins of prog_topic, relevant_snippets and user are necessary, to access user_email

No joins (lookups) are necessary, as we embedded user_email and prog_language into snippets

```
1 --- Initial filtering for snippets with given lang and timestamp
2 WITH relevant_snippets AS (
3   SELECT id, prog_topic_id
4   FROM prog_snippet
5   WHERE prog_language_id = ${progLanguageId}
6   AND created_at BETWEEN (now() - INTERVAL '1 month') AND now()
7 ),
8 --- After filter, perform join and aggregation to access the relevant user info
9 report_result AS (
10  SELECT "user".email AS user_email
11  FROM prog_topic
12  JOIN "user" ON "user".id = prog_topic.user_id
13  JOIN relevant_snippets ON relevant_snippets.prog_topic_id = prog_topic.id
14  GROUP BY user_email
15  HAVING COUNT(relevant_snippets.id) >= 3
16 )
17 --- Order the result set by the email address
18 SELECT user_email
19 FROM report_result
20 ORDER BY user_email;
```

```
1 db.prog_snippet.aggregate([
2   // 1. Keep only the relevant fields, get the creation date as ISO string
3   {
4     $project: {
5       _id: 1,
6       prog_language_name: "$prog_language.name",
7       prog_snippet_created_at: {
8         $convert: { input: "$created_at", to: "string" }
9       },
10      user_email: 1
11    }
12  },
13  // 2. Keep only the relevant snippets
14  {
15    $match: {
16      $expr: {
17        $and: [
18          { $eq: ["$prog_language_name", progLanguageName] },
19          { $gte: ["$prog_snippet_created_at", oneMonthAgo] },
20          { $lte: ["$prog_snippet_created_at", now] }
21        ]
22      }
23    }
24  },
25  // 3. Group by email, assign it to doc_id, as it is unique at this stage
26  { $group: { _id: "$user_email", count: { $sum: 1 } } },
27  // 4. Keep only those emails where the snippet count is at least 3
28  { $match: { $expr: { $gte: ["$count", 3] } } },
29  // 5. Keep only the email field, without the count
30  { $project: { _id: 0, email: "$_id" } },
31  // 6. Sort by email
32  { $sort: { email: 1 } }
33 ]);
```

Query Comparison - **Dominant Languages**

```
1 --- 2. Select snippets that are in relevant topics
2 --- and create a column for the length of their content
3 WITH relevant_snippets AS (
4     SELECT prog_language_id, LENGTH(content) as local_len
5     FROM prog_snippet
6     WHERE prog_topic_id IN
7 --- 1. Select relevant topics for given tag id
8     (SELECT progTopicId
9      FROM tags_on_prog_topics
10     WHERE tagId = ${tagId}))
11 --- 3. Join with prog language and group snippets by language
12 --- create an aggregate column with the sum of the grouped snippets
13 --- Order by the length and take the first 10
14 SELECT name, version, SUM(local_len) as length
15 FROM relevant_snippets
16 JOIN prog_language
17     ON prog_language.id = relevant_snippets.prog_language_id
18 GROUP BY prog_language_id, name, version
19 ORDER BY length DESC
20 LIMIT 10
```

- **SQL:** Join with prog_language necessary to obtain name & version
- **NoSQL:** NOT necessary (embedded)
- Lookup of prog_snippet still needed in **NoSQL** (too large to embed)

Query Comparison - Dominant Languages

```
1 db.progTopic.aggregate([
2   {
3     // 1 .filter topics with the given tag name
4     $match: {
5       tags: {
6         $elemMatch: { name: tagName },
7       },
8     },
9   },
10  {
11    // 2. only keep snippet id arrays
12    $project: {
13      _id: 0,
14      prog_snippet_ids: 1,
15    },
16  },
17  {
18    // 3. join snippets that are associated with the topics
19    // as prog_snippet_ids is an array, prog_snippets is also an array (of snippets)
20    $lookup: {
21      from: 'prog_snippet',
22      localField: 'prog_snippet_ids',
23      foreignField: '_id',
24      as: 'prog_snippets',
25    },
26  },
27  {
28    // 4. only keep snippet arrays
29    $project: {
30      _id: 0,
31      prog_snippets: 1,
32    },
33  },
34  {
35    // 5. create a document for each snippet
36    $unwind: '$prog_snippets',
37  },
```

```
38  {
39    // 6. keep only the relevant fields
40    $project: {
41      lang: '$prog_snippets.prog_language',
42      content: '$prog_snippets.content',
43    },
44  },
45  {
46    // 7. group by language
47    // create an aggregate length field with the sum of the length of the content
48    $group: {
49      _id: '$lang',
50      length: {
51        $sum: { $strLenCP: '$content' },
52      },
53    },
54  },
55  {
56    // 8. map fields to create wanted output format
57    $project: {
58      _id: 0,
59      name: '$_id.name',
60      version: '$_id.version',
61      length: 1,
62    },
63  },
64  {
65    // 9. sort by length descending and name ascending
66    $sort: {
67      length: -1,
68      name: 1,
69    },
70  },
71  {
72    // 10. only take the first 10 results
73    $limit: 10,
74  },
75 ],);
```



Thank You
for Your Attention

