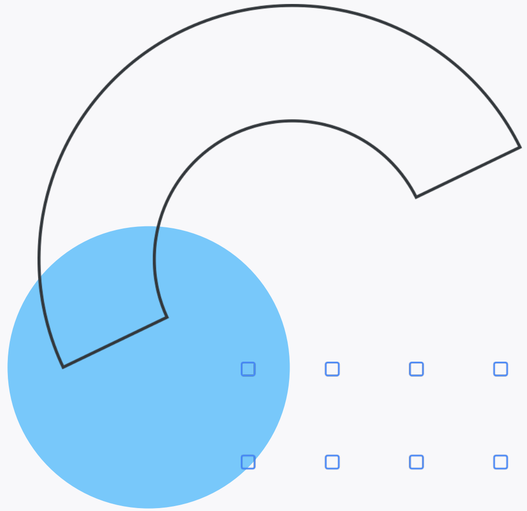
Databend

Databend JSON 的设计与实现

主讲人：白珅

2022.10

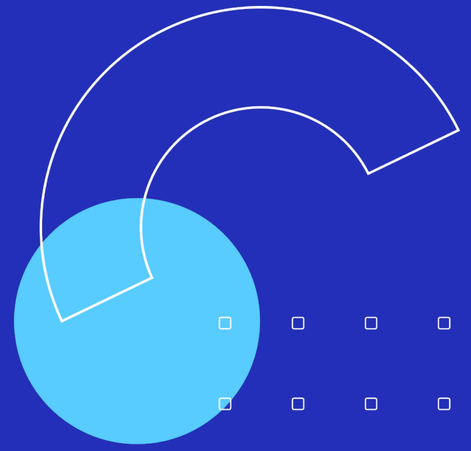


Data infra 研究社

自我介绍

Databend

- 白珅
  - Datafuselabs 数据库开发工程师



Databend

目录

CONTENTS

- Databend JSON 使用介绍
- jsonb 编码结构介绍
- jsonb 性能分析

Databend

Databend JSON 使用介绍

Data infra 研究社

为什么使用 JSON？

Databend

日常工作中，大量接口导入的数据是半结构化的 JSON 数据

- 字段多而且复杂
- 数据存在嵌套关系
- 同一个字段的数据类型可能不同


```
{
  "id":"24338586768",
  "type":"PushEvent",
  "actor":{
    "id":98194605,
    "login":"FeNjK",
    "display_login":"FeNjK",
    "gravatar_id":"",
    "url":"https://api.github.com/users/FeNjK",
    "avatar_url":"https://avatars.githubusercontent.com/u/98194605?
    "
  },
  "repo":{
    "id":541301499,
    "name":"FeNjK/react-mesto-api-full",
    "url":"https://api.github.com/repos/FeNjK/react-mesto-api-full"
  },
  "payload":{
    "push_id":11194684964,
    "size":1,
    "distinct_size":1,
    "ref":"refs/heads/main",
    "head":"d6f345990d263c8d0553cefaa117d8a3235de5f7",
    "before":"dd1f642957c98f68058f1063d05c693181b3faaa",
    "commits":[
      {
        "sha":"d6f345990d263c8d0553cefaa117d8a3235de5f7",
        "author":{
          "email":"bodhisatva_xp@mail.ru",
          "name":"FeNjK"
        },
        "message":"fix: исправил доменные имена в файле
        README.md",
        "distinct":true,
        "url":"https://api.github.com/repos/FeNjK/react-mesto-
        api-
```


数据举例：来自 github 事件归档数据https://gharchive.org/

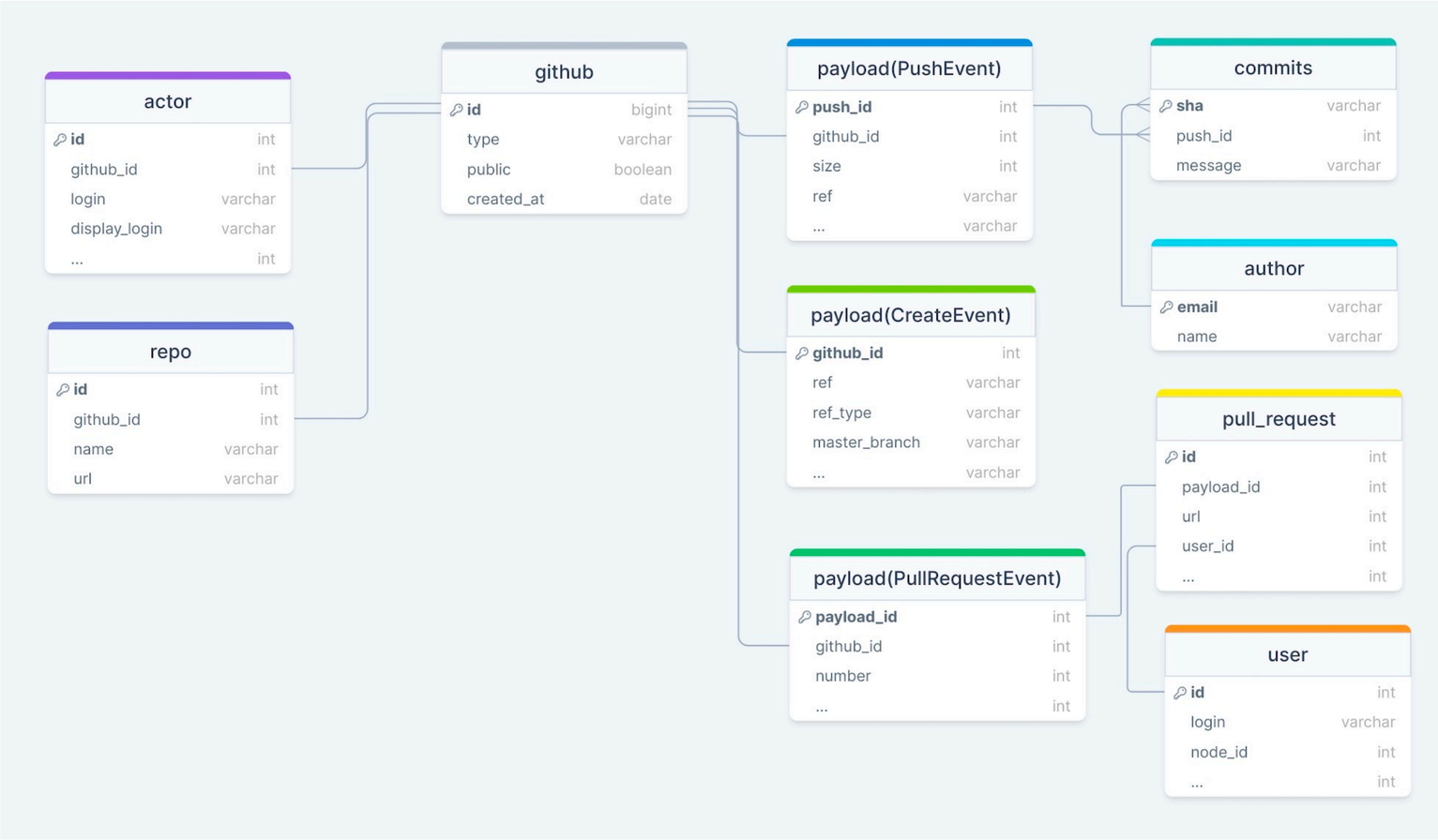
为什么使用 JSON？

Databend

如果没有JSON数据类型

- 嵌套数据需创建多张表
- 新增字段需修改表结构
- 嵌套字段查询需要复杂的 join

示例：一个没有JSON数据类型的半结构化数据表结构



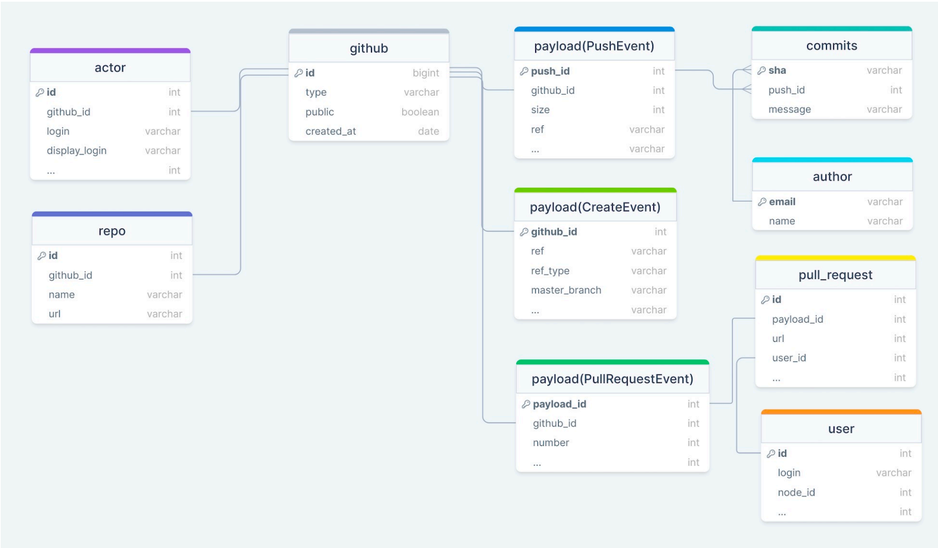
为什么使用 JSON？

Databend

如果没有JSON数据类型

- 嵌套数据需创建多张表
- 新增字段需修改表结构
- 嵌套字段查询需要复杂的 join

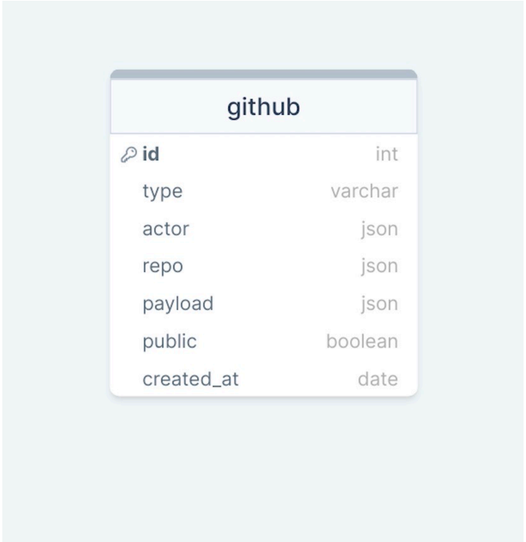
示例：一个没有JSON数据类型的半结构化数据表结构



使用JSON数据类型后

- 无论多少嵌套层级，仅需1张表
- 新增字段无需修改表结构
- 方便查询多层嵌套的字段

示例：一个使用Json数据类型的非结构化数据表结构



Databend中JSON 数据操作

Databend

Databend将JSON数据类型也叫做Variant，目前我们支持JSON数据的**导入**和**查询**

- **导入流程**

创建表 → 导入

```
CREATE TABLE github_2022101012 (
  `id` string,
  `type` string,
  `actor` json,
  `repo` json,
  `payload` json,
  `public` boolean,
  `created_at` timestamp
);
```

方式一：INSERT

```
INSERT INTO
  github_2022101012
VALUES
  (
    '123',
    'test',
    parse_json('{"id":1}'),
    parse_json('{"name":"abc"}'),
    parse_json('[1,2,3]'),
    true,
    '2022-10-20 00:00:00'
  );
```

方式二：COPY

```
COPY INTO github_2022101012
FROM
  'https://data.gharchive.org/2022-10-10-12.json.gz'
  FILE_FORMAT = (TYPE = NDJSON COMPRESSION = AUTO);
```

方式三：接口

```
curl -H "insert_sql:insert into github_2022101012 format NdJson"
     -H "format_skip_header:0"
     -F "upload=@/tmp/github_2022101012.ndjson"
     -u root:test
     -XPUT "http://localhost:7000/v1/streaming_load"
```

Databend中JSON 数据操作

Databend

- **查询流程**

目前支持下列操作符查询嵌套数据：

操作符	说明	示例
[index]	按索引查询 JSON Array	SELECT parse_json('[[1,2],[3,4]]')[0][1];
['field']	按字段名查询 JSON Object	SELECT parse_json('{"k":"v"}')['k'];
:field		SELECT parse_json('{"k":"v"}'):k;
.field		SELECT parse_json('{"k1":{"k2":"v"}}'):k1.k2;

```
SELECT id, actor:login, repo['name'], payload:commits[0]:author.name
FROM
  github_2022101012
ORDER BY id ASC LIMIT 2;

+-------------+--------------------+-----------------------------------+-------------------------------
| id          | actor:login        | repo['name']                      | payload:commits[0]:author.name
+-------------+--------------------+-----------------------------------+-------------------------------
| 24499958028 | "YubaC"            | "YubaC/2810security.github.io"    | "YubaC"
| 24499958046 | "ValentinaZussino" | "ValentinaZussino/js-fizzbuzzdom" | "ValentinaZussino"
+-------------+--------------------+-----------------------------------+-------------------------------
```

JSON 实现方式

Databend

Databend采用 jsonb 作为 JSON 的实现方式

JSON实现方式对比

	文本（JSON）	二进制（jsonb）
解析速度	慢	快
嵌套字段查询速度	慢	快
支持扩展数据类型	否	是
占用空间	正常	稍大

Databend

jsonb 编码结构介绍

Data infra 研究社

jsonb 介绍

Databend

jsonb (json binary/better) 是 PostgreSQL 设计的一种二进制 JSON 数据格式，2014年 9.4 版本发布

- JSON查询性能大幅提升
- 支持 NoSQL 的关系型数据库
- 用户最喜欢的PostgreSQL三个特性之一

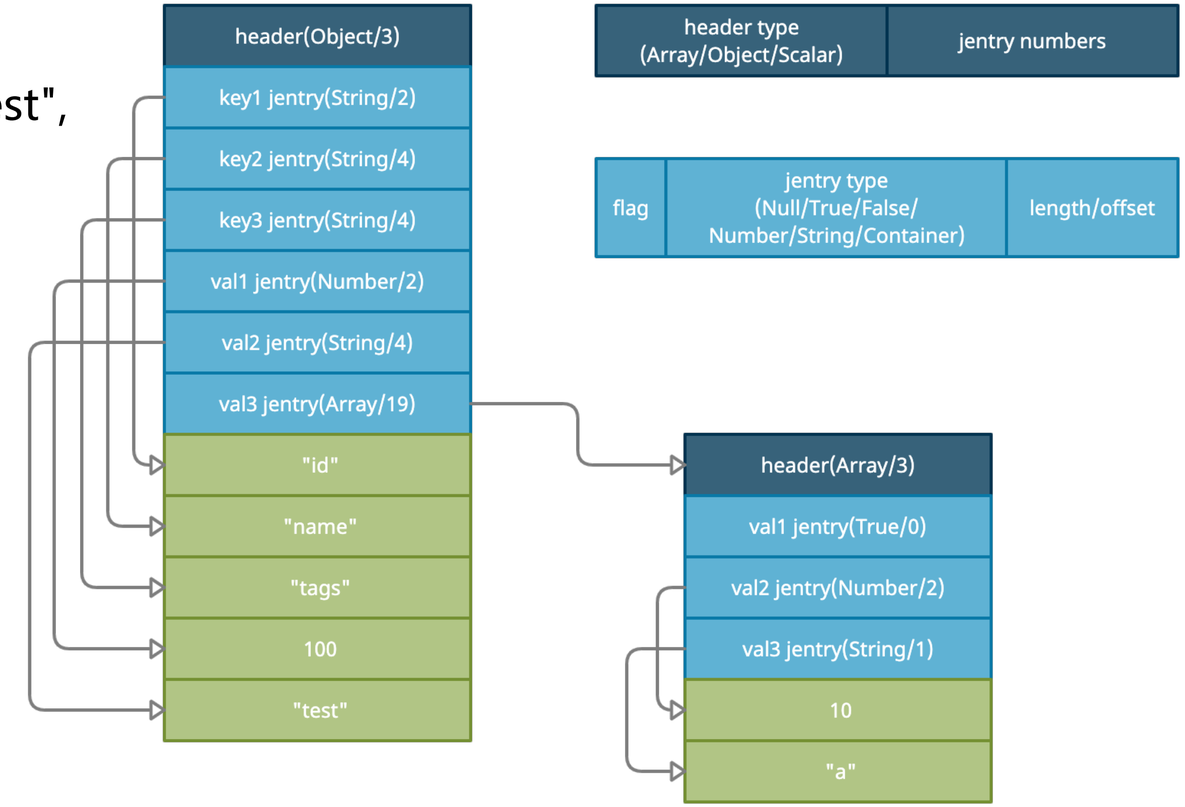
http://www.sai.msu.su/~megera/postgres/talks/jsonb-pgvision-2021.pdf

jsonb 数据格式

Databend

jsonb数据格式包含3部分：header，jentry，数据

```
{
    "id": 100,
    "name": "test",
    "tags": [
        true,
        10,
        "a"
    ]
}
```



header(4 bytes)

记录数据类型和 jentry 的个数

1. 嵌套类型（Array/Object），内部包含多条数据
2. 标量类型（Scalar），只有一条数据

jentry(4 bytes)

记录内部数据类型和数据偏移位置

1. flag，标识两种寻址方式
2. type (Null/True/False/Number/String/Container)
3. offset/length，flag标识的两种寻址方式，offset 查找更快，但不利于压缩，length 相反

数据

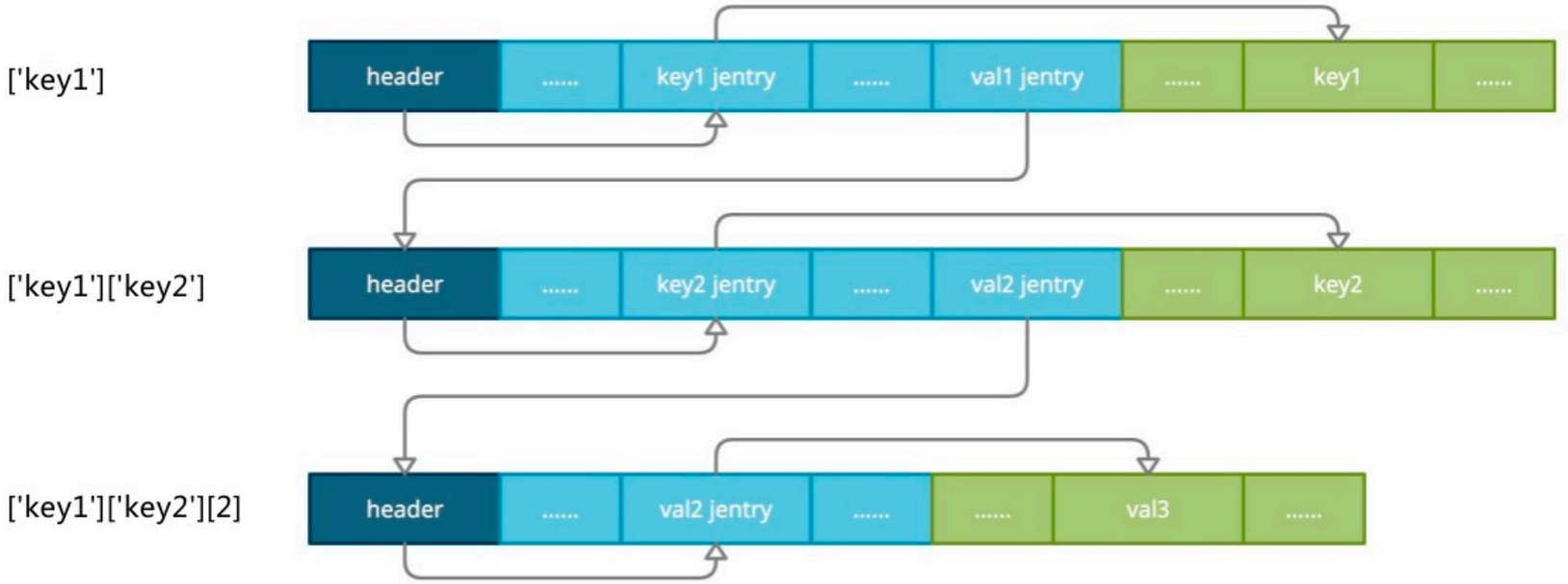
实际存储的数据，包括数字和字符串

jsonb 数据查询流程

Databend

通过 jentry 的记录直接定位数据位置， 加快查询速度

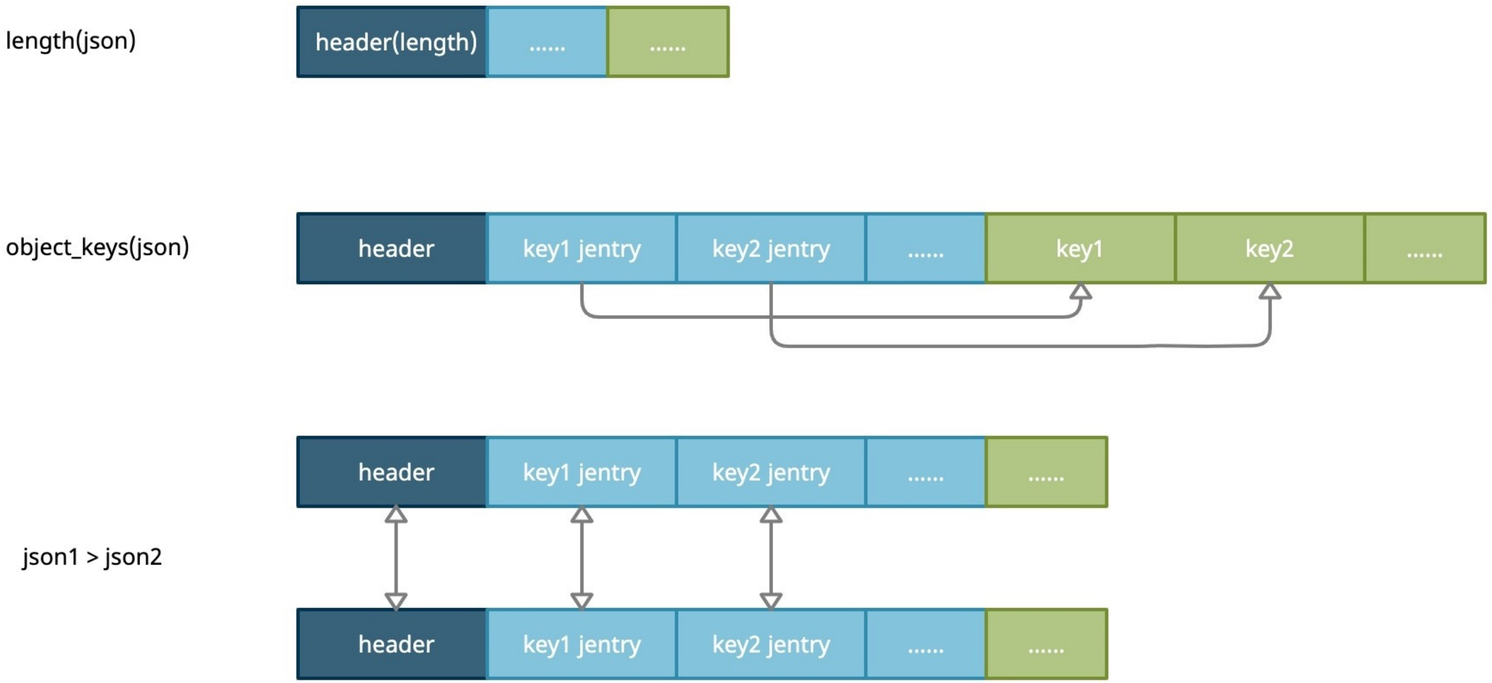
```
{
  "key1":{
    "key2":[
      1,
      2,
      3
    ]
  }
}

SELECT
v['key1']['key2'][2]
```



jsonb 函数计算

Databend

jsonb 的函数可以通过 header 和 jentry 中记录的相关信息直接获取到结果，不需要解析全部数据，可以加快查询速度并减少内存分配



Databend如何实现jsonb ？

Databend

Databend 最初是使用文本格式来实现JSON，为了优化性能，采用 PostgreSQL 的设计实现了 Rust 版本的 jsonb

- parser 将 JSON 字符串转为 jsonb Value
- 二进制 jsonb 数据的编码和解码功能
- 基于二进制数据操作的函数，支持部分解析

  函数相关文档可查阅：https://databend.rs/doc/reference/functions/semi-structured-functions
- 同时支持对 JSON 和 jsonb 的操作，兼容旧版数据

Databend实现jsonb的代码地址：
https://github.com/datafuselabs/databend/tree/main/src/common/jsonb

欢迎感兴趣的同学共同参与！

Databend

jsonb 性能分析

Data infra 研究社

测试数据

Databend

使用 Github 2022-10-10-12 的事件数据作为测试数据集

https://data.gharchive.org/2022-10-10-12.json.gz

- 共有 219445 行数据
- 包括 actor，repo，payload 等嵌套结构，其中 payload 最多5层嵌套结构

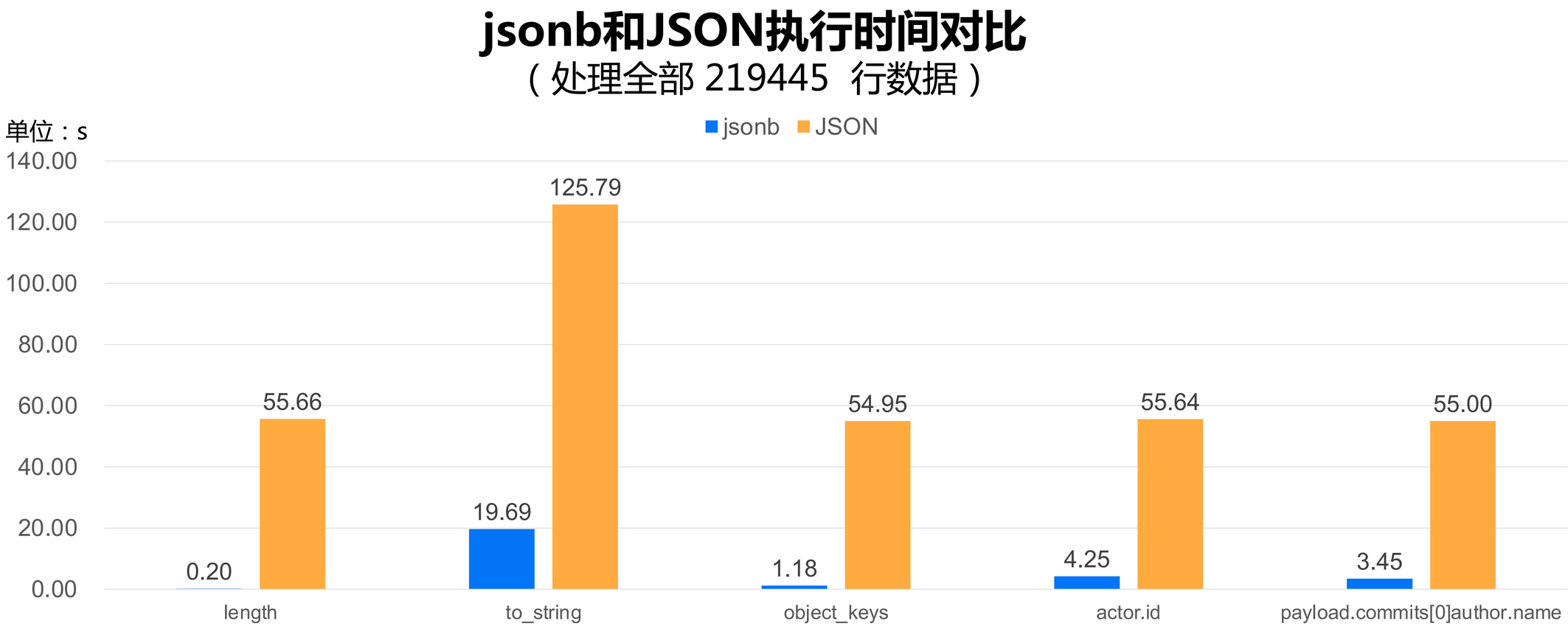
测试函数（对比 jsonb 和 serde_json 的性能）

1. length() 获取 JSON 数组的长度
2. to_string() 将 JSON 作为一个整体查询时使用
3. object_keys() 获取 JSON Object 的所有 keys
4. actor.id 查询嵌套在 actor 内部的 id
5. payload.commits[0]author.name 查询 payload 内部嵌套的 name

执行时间

Databend

使用jsonb数据格式较使用JSON，执行时间指标均大幅下降

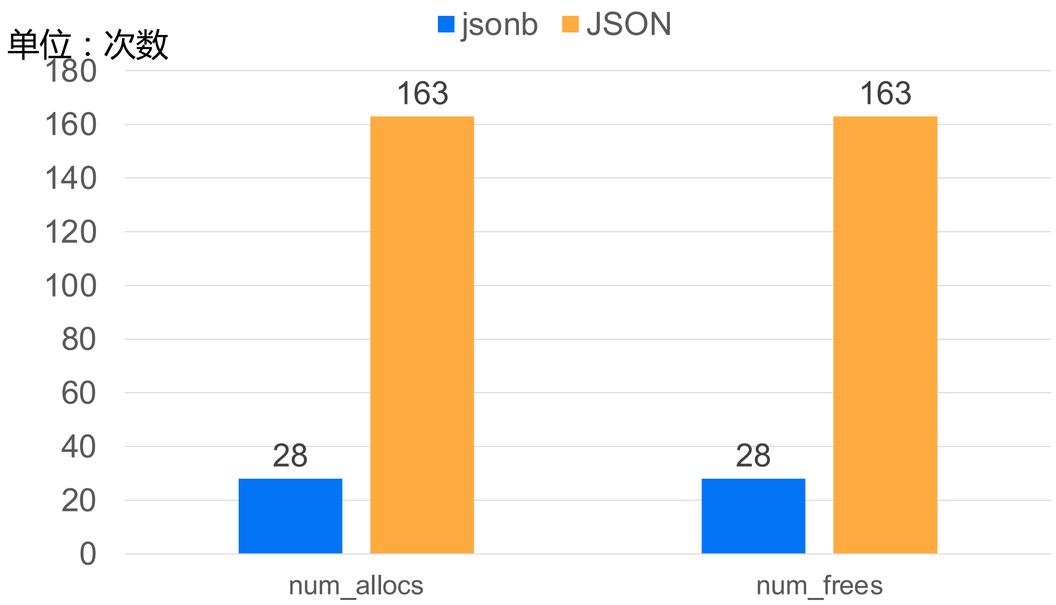


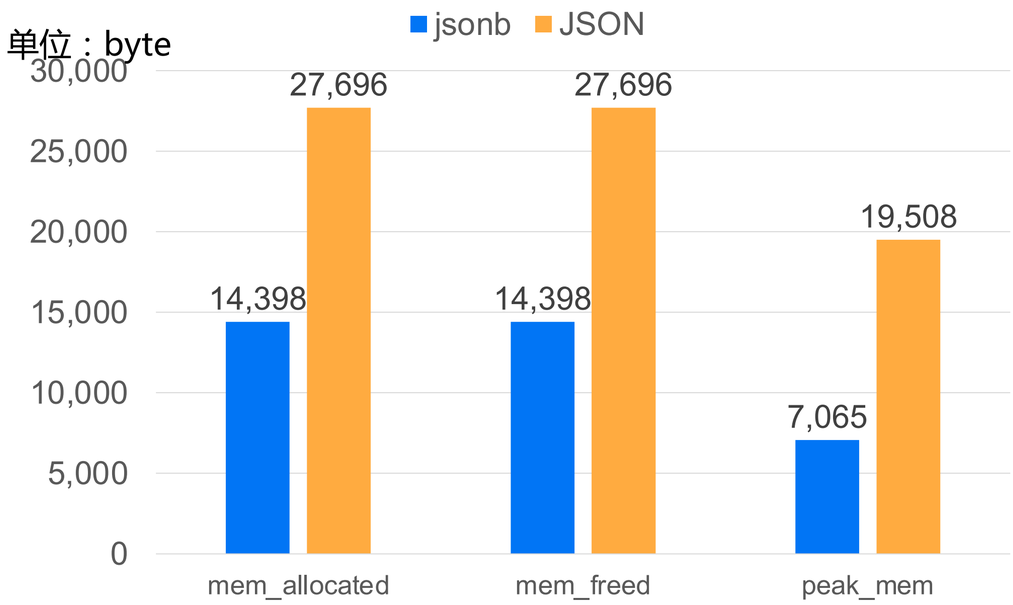
占用的内存

Databend

使用jsonb数据格式较使用JSON，内存分配次数减少 80%+，占用内存下降50%+

Jsonb和JSON执行 to_string() 函数占用内存对比





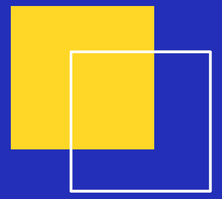
下一步规划

Databend

- 完成表达式迁移，替换文本 JSON 类型
- 优化数据解析速度
- 支持更多的 JSON 函数
- 扩展数据类型（Date, Timestamp)
- 支持虚拟列，进一步提高查询速度

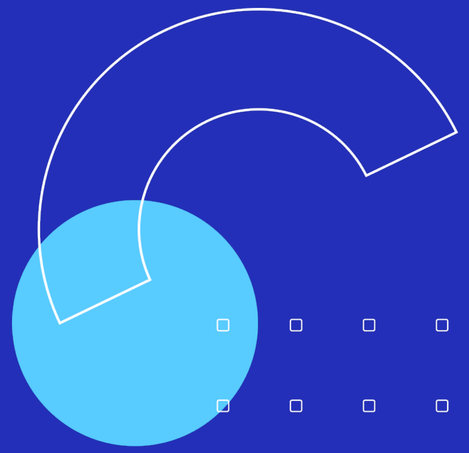
Databend

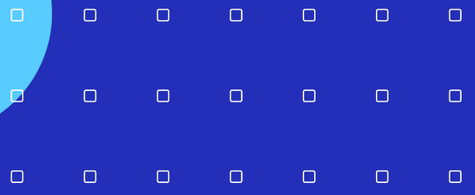
Q & A



Databend

THANKS!





Data infra 研究社