NATIVE JSON DATA TYPE IN MYSQL

ABOUT ME

- Franz Wilding
- Full-Stack Dev & Founder of  unite
- *radical flexible* cms: unitecms.io

WHAT IS THIS ABOUT?

- Introduction of native JSON data type in MySQL
- Discussion if / when we should use it
- *(Most of the stuff is also valid for PostgrSQL)*

IN A NUTSHELL

- Introduced in MySQL 5.7
- Native JSON data type
- JSON functions for text and JSON data types
- Shorthand syntax for querying data

DIFFERENCES TO TEXT TYPE

- automatic **data validation**
- optimized storage
  - binary format
  - no string parsing of the whole document required
  - quick read access
  - quick nested lookup

PERFORMANCE EXAMPLE

```
SELECT DISTINCT feature->"$.type" as json_extract FROM t;
1 row in set (1.25 sec)



ALTER TABLE t CHANGE feature feature LONGTEXT NOT NULL;

SELECT DISTINCT feature->"$.type" as json_extract FROM t;
1 row in set (12.85 sec)
```

(~200k rows)

https://mysqlserverteam.com/taking-the-new-mysql-5-7-json-features-for-a-test-drive/

SOME EXAMPLES

BUT WAIT, WHAT ABOUT NF1?

A relation is in first normal form if and only if the domain of each attribute contains only ***atomic*** *(indivisible) values, and the value of each attribute contains only* ***a single value*** *from that domain.*

WHAT DOES ATOMIC / INDIVISIBLE MEAN?

- Definition: *"cannot be decomposed into smaller pieces by the DBMS"*
- Problem: Even strings can be divided into substrings

CHRIS DATE'S DEFINITION

Every row-and-column intersection contains exactly one value from the applicable domain (and nothing else)

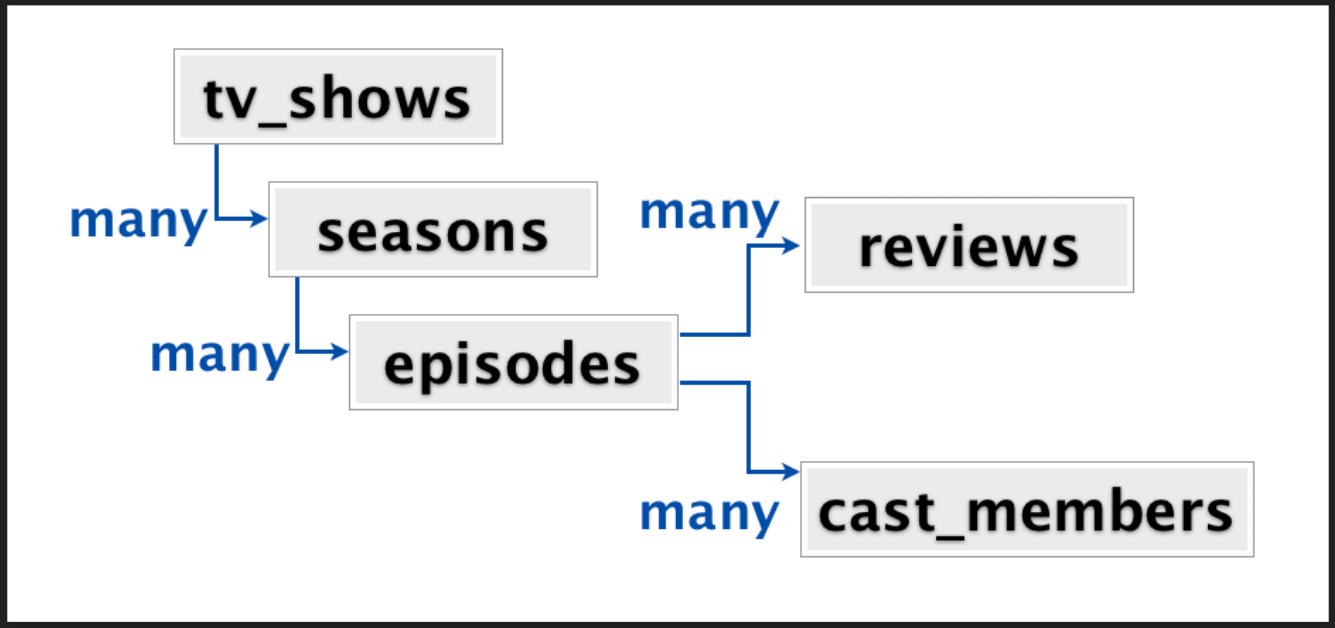
- What about HTML in a column?
- A geometric shape in JSON?
- A log file?

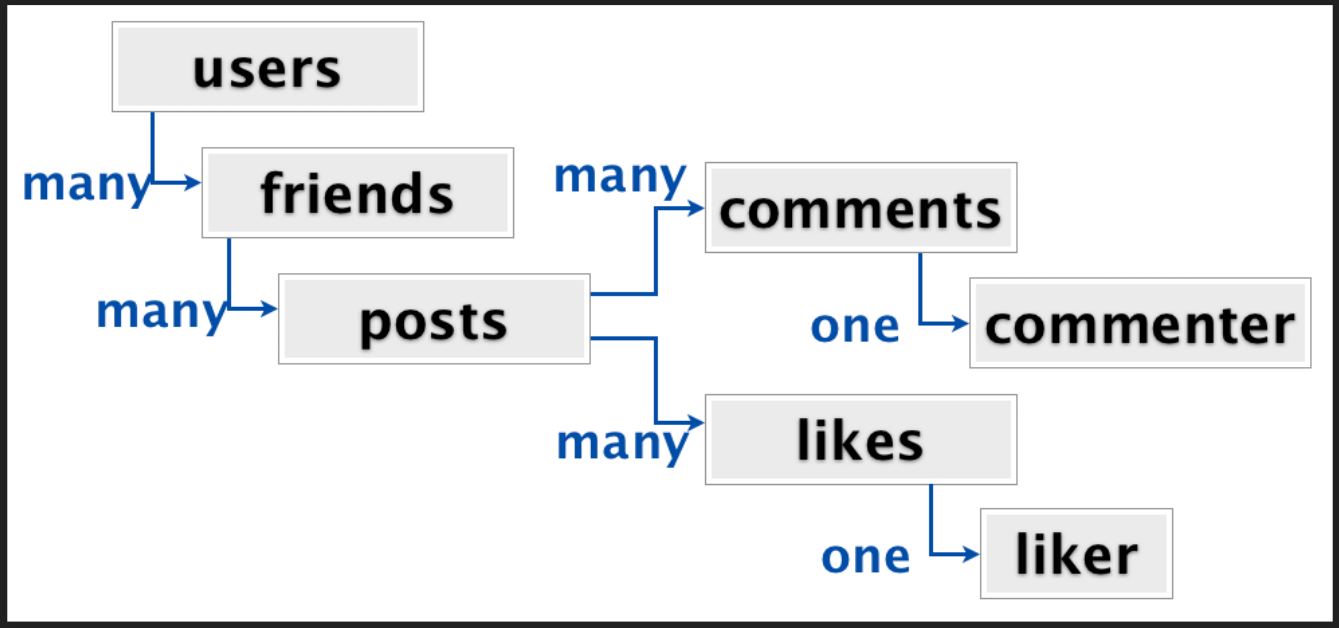
Answer: It depends.

DO YOU CARE ABOUT THE INNER STRUCTURE OF THE JSON DOCUMENT?

SARAH MEI: WHY YOU SHOULD NEVER USE MONGODB (2013)

The ideal case for MongoDB:





Seven-table joins. Ugh. Suddenly storing each user's activity stream as one big denormalized nested data structure, rather than doing that join every time, seems pretty attractive.

WHAT COULD POSSIBLY GO WRONG?

- Shared types: A user could be a friend could be a commenter could be a liker
- No need for JOINs **BUT** high risk of inconsistent

```
{
 name: Joe,
 url: '...',
 stream:
 [{
   user: {name: Jane, url: '...'},
   title: 'today',
   body: 'go fly a kite',
   likes: [
    {user: {name: Lu, url: '...'}},
    {user: {name: Joe, url: '...'}}
   ],
 }]
}
```

```
{id: 1,
 name: Joe,
 url: '...',
 stream:
 [{
   user: 2,
   title: 'today',
   body: 'go fly a kite',
   likes: [
    {user: 3},
    {user: 1},
   ],
 }]
}
```

Once we started doing ugly MongoDB joins manually in the Diaspora code, we knew it was the first sign of trouble. It was a sign that our data was actually relational, that there was value to that structure, and that we were going against the basic concept of a document data store.

THEY REALIZED, THEY WHERE USING A CACHE INSTEAD OF A DB

- 8 months of production data, ~ 1.2 million SQL rows
- Moving (inconsistent) MongoDB data to MySQL: https://speakerdeck.com/sarahmei/switching-data-stores-a-postmodern-comedy

LEARNING

The only thing MongoDB is good at is storing arbitrary

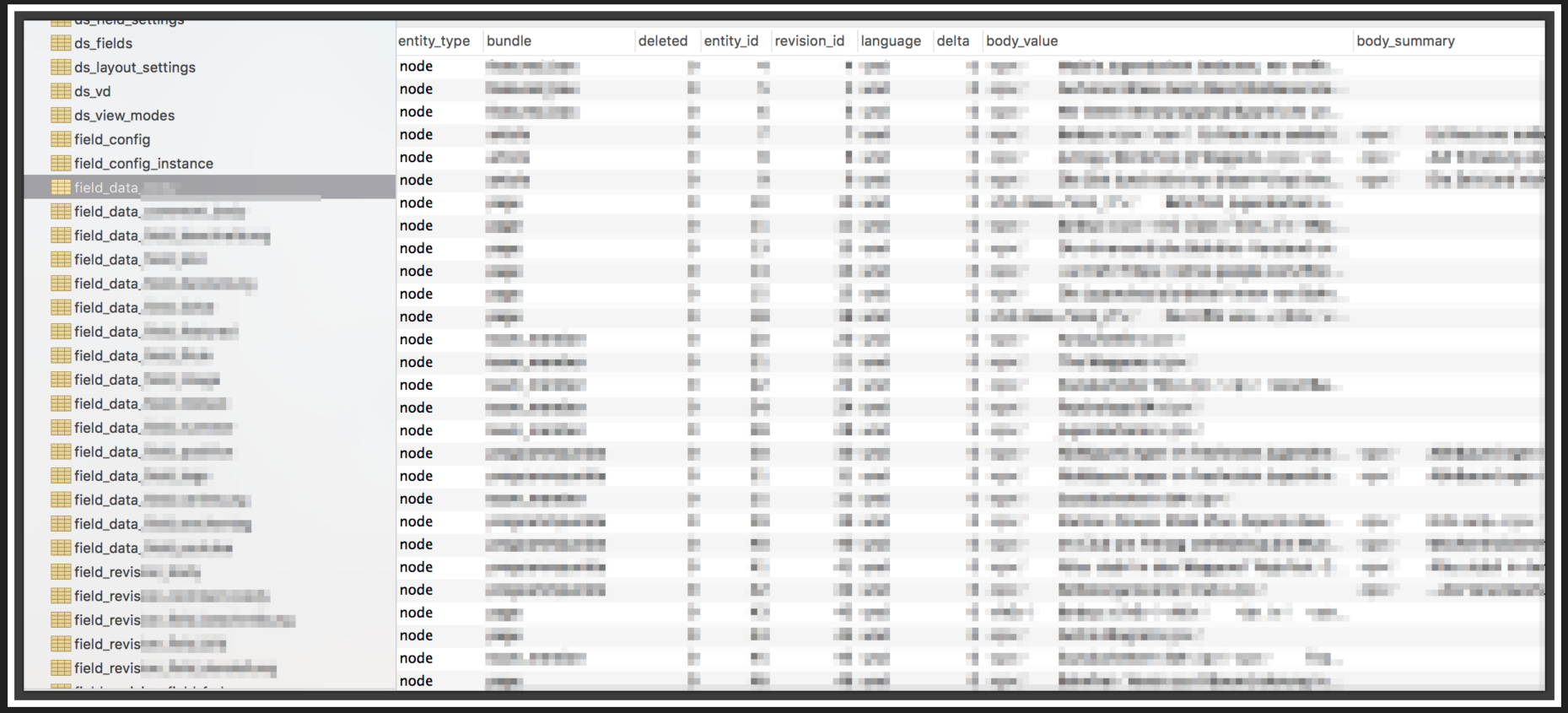
– you don’t care at all what’s inside –

pieces of JSON

OUR USE CASE: DYNAMIC CONTENT SCHEMAS

- Allow to define data structures dynamically on application level
- The user can change the structure without the need to update any code

THE DRUPAL WAY



THE DRUPAL WAY

- Drupal is doing it with a table for each field
- CREATE / UPDATE / DELETE fields modifies db schema
- (most) field tables only have a single value + meta infos
- JOIN over **many** tables to get to the value

THE DRUPAL WAY

```
  $result = db_query("SELECT [illegible] FROM node
INNER JOIN field_data_field_[illegible] AS [illegible] ON ([illegible] = node.nid)
INNER JOIN field_data_field_[illegible] ON node.nid = [illegible] AND [illegible] = 'node'
LEFT JOIN field_data_field_[illegible] as [illegible] ON node.nid = [illegible] AND [illegible] = 'node'
LEFT JOIN field_data_field_[illegible] as [illegible] ON node.nid = [illegible] AND [illegible] = 'r
WHERE
node.status = 1
AND [illegible] IS NULL OR [illegible] != 'push')
AND [illegible] IS NULL OR [illegible]
AND node.type IN [illegible]
AND [illegible]
AND(
        SELECT [illegible]
    [illegible]
    [illegible]
    [illegible]
      [illegible]
      [illegible]
      [illegible]

) = 0

AND(
        SELECT [illegible]
    [illegible]
    [illegible]
    [illegible]
      [illegible]
      [illegible]
      [illegible]

) < 3

AND(
        SELECT [illegible]
    [illegible]
    [illegible]
    [illegible]
      [illegible]
      [illegible]
      [illegible]

) < 3
```

UNITE CMS

- db schema is in sync with doctrine entities
- does only change on code change
- A row for each content item with one json field for the data
- For the database, the json content is atomic
- The application is responsible for the inner structure

UNITE CMS

id	conte...	data	created	updated
00766c2a-2c32-11e8-881d-6308fcd844ce	2	{"title": [redacted], "client": [redacted] "website": [redacted], "descriptio...	2018-03-20 11:29:50	2018-03-20 11:40:52
3be4bda8-2c31-11e8-881d-6308fcd844ce	1	{"file": {"id": "63d1aec2-332d-11e8-a9ca-80e65026b996", "name": [redacted]	2018-03-20 11:24:20	2018-03-29 08:45:32
433ecaf8-2c31-11e8-881d-6308fcd844ce	1	{"name": "Offer", "weight": 1}	2018-03-20 11:24:32	2018-03-20 11:26:09
494826e2-2c31-11e8-881d-6308fcd844ce	1	{"name": "CMS", "weight": 2}	2018-03-20 11:24:42	2018-03-20 11:43:49
7eadafc6-2db9-11e8-a54a-e6728649db71	1	{"file": null, "name": "CMS", "image": null, "weight": 1}	2018-03-22 10:12:15	2018-03-29 08:37:36
8bdcab1e-2dac-11e8-a54a-e6728649db71	2	{"title": [redacted], "client": null, "website": null, "description": null}	2018-03-22 08:39:33	2018-03-22 08:39:33
8e2f38dc-2dac-11e8-a54a-e6728649db71	2	{"title": [redacted], "client": null, "website": null, "description": null}	2018-03-22 08:39:37	2018-03-22 08:39:37
984b4208-2db0-11e8-a54a-e6728649db71	2	{"title": null, "client": null, "website": null, "description": null}	2018-03-22 09:08:32	2018-03-22 09:08:32
d2ace6e4-3824-11e8-9206-cf8017ef6c4e	3	{"fistname": [redacted], "lastname": [redacted]}	2018-04-04 16:25:43	2018-04-04 16:25:43
d5e54a18-3824-11e8-9206-cf8017ef6c4e	3	{"fistname": [redacted], "lastname": [redacted]}	2018-04-04 16:25:49	2018-04-04 16:25:49

TECHNICAL PROSPECTIVE

- MySQL 8
  - New JSON functions
  - Performance optimizations
  - partial, in-place updates of JSON column values

THANK YOU!

franz@unite.co.at

unitecms.io