Из бэкенда* в дата-инженерию: типовые ошибки и паттерны**

* и не только

**по субъективному мнению автора

```
>>> from dateutil.parser import parse
>>> parse("23.08.2024 21:04:53")
datetime.datetime(2024, 8, 23, 21, 4, 53)
>>> parse("12.08.2024 21:04:53")
datetime.datetime(2024, 12, 8, 21, 4, 53)
```

:: спикер

Photo by Gema Saputera on Unsplash

Юля Волкова

7+ лет дата/бэкенд инженерии, 15+ в / около ИТ

https://github.com/xnuinside

Data Engineering
Lead at >>>

CODE SCORING

* о причинах доклада (прямо сейчас и поговорим)

* как понять что вы по уши в дата-инженерии

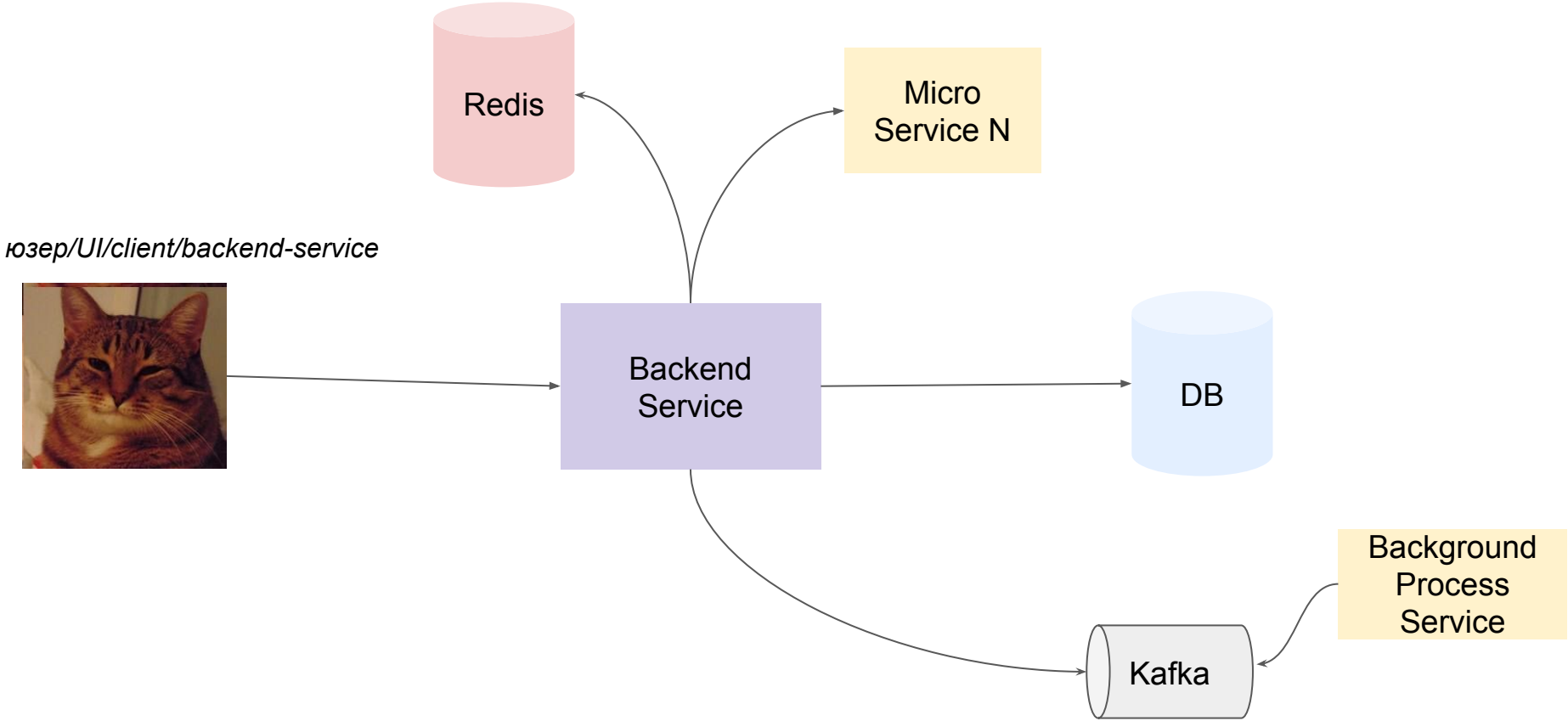
~~* хорошие паттерны и best-practice бэкенда~~

просто какие-то паттерны

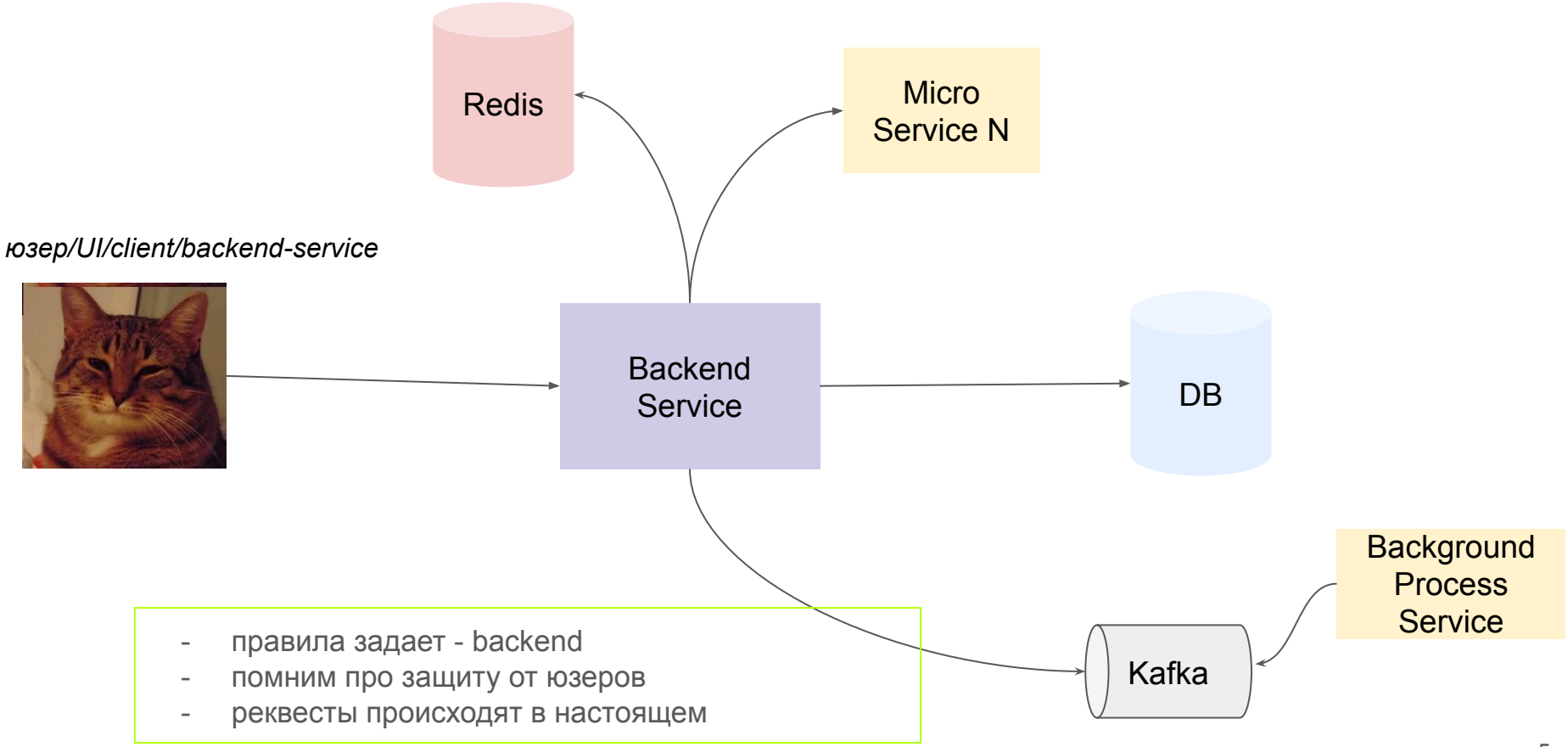
* немного data-теории

* выводы

:: пример backend-архитектуры



:: пример backend-архитектуры

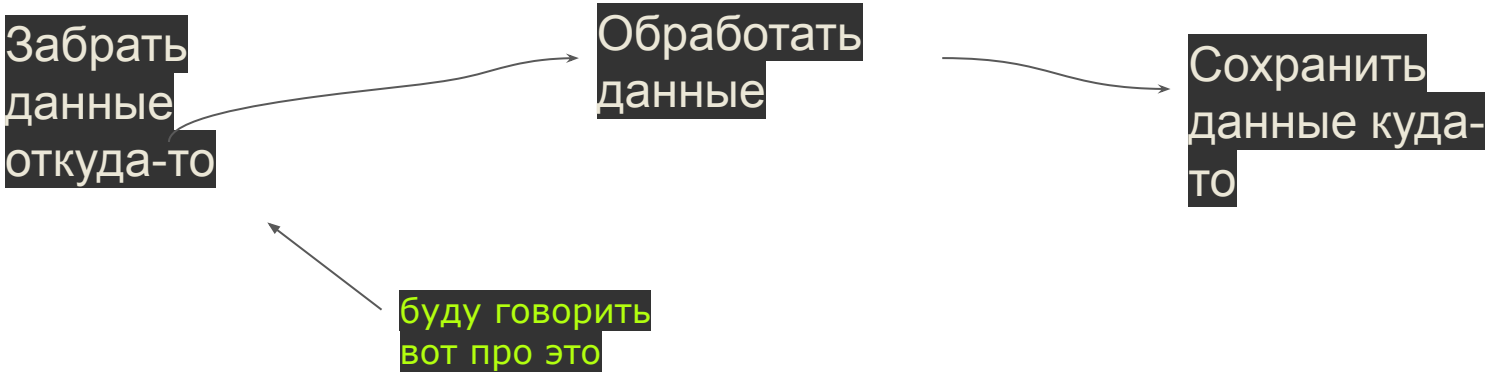


- какой-то web-framework - Django / Flask / FastAPI / aiohttp и тд
- какая-то ORM - Django ORM / SQLAlchemy / TortoiseORM / GinoORM / PonyORM / Peewee ORM/ ChtoUgodnoORM
- какие-нибудь сериализаторы - Pydantic, Marshmallow, msgspec и тд
- клиенты в redis / kafka (можно даже и async)
- httpx/request/orjson/другие утилитные библиотеки для реквестов/форматов данных

создает регулярные процессы по работе с массивами данных

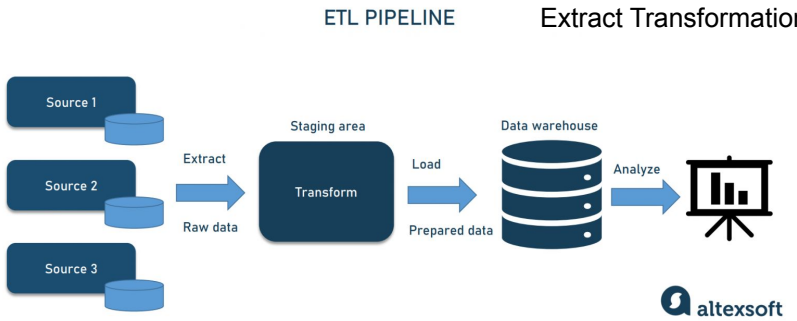
а эти данные кто-то нам отдает

эти данные кто-то потребляет

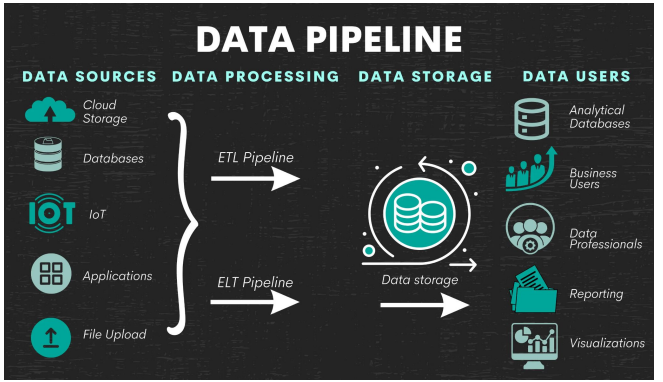


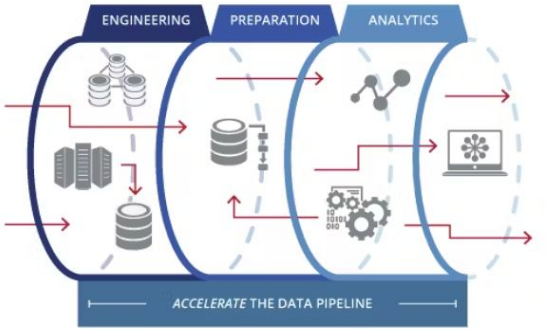
Как понять, что возле вас запахло Data-инженерией?

:: если погуглить чем занимается дата-инженер



https://habr.com/ru/articles/743308/





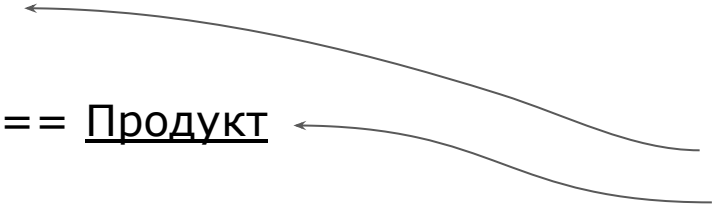
https://www.mongodb.com/resources/basics/data-engineering

https://www.stratascratch.com/blog/what-does-a-data-engineer-do-and-what-they-dont-do/

* фичи Продукта базируются на данных

* сами данные == Продукт

* данные для внутренней аналитики / DS

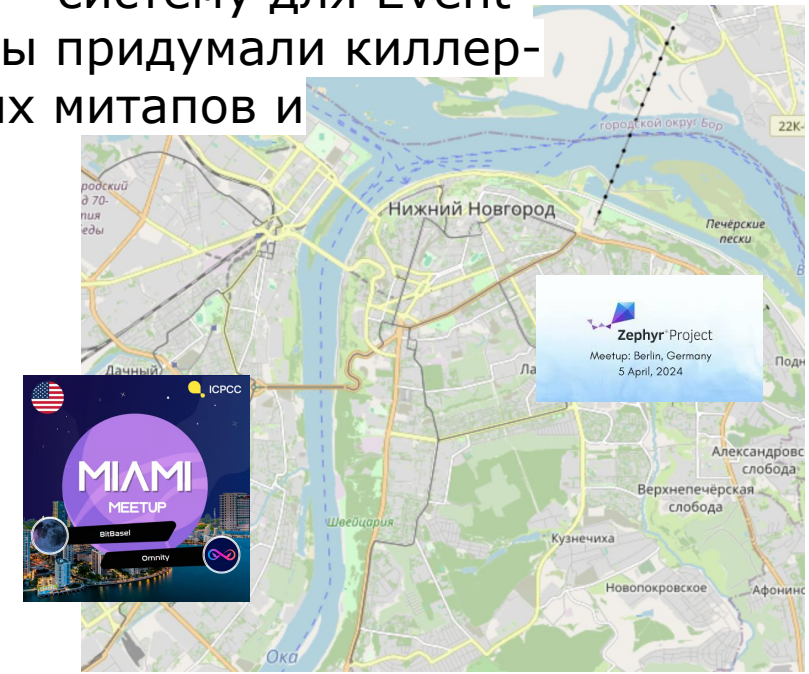


имеется в виду то, что приносит деньги компании

У вас появляется **РЕГУЛЯРНЫЙ** процесс по доставке массивов данных из **точки А** в **точку Б** (с трансформациями / без) - это дата инженерия

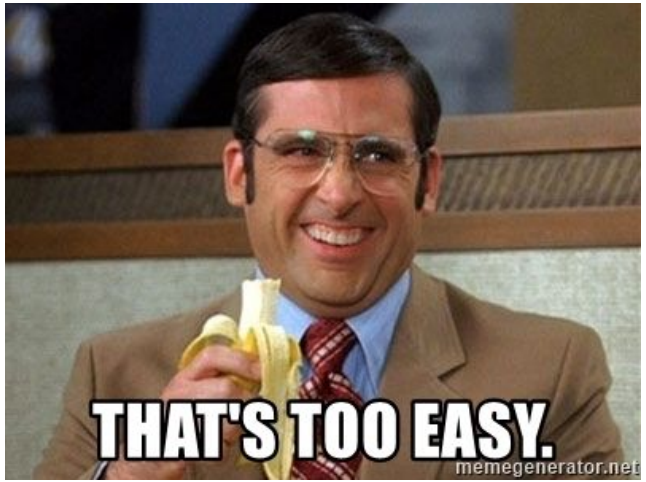
Допустим, я пилю Продукт - систему для Event-мейкеров и DevRel-ов. И мы придумали киллер-фичу - базу всех доступных митапов и конференций в регионе.

- поиск по категориям
- датам
- локациям
- и другое волшебство

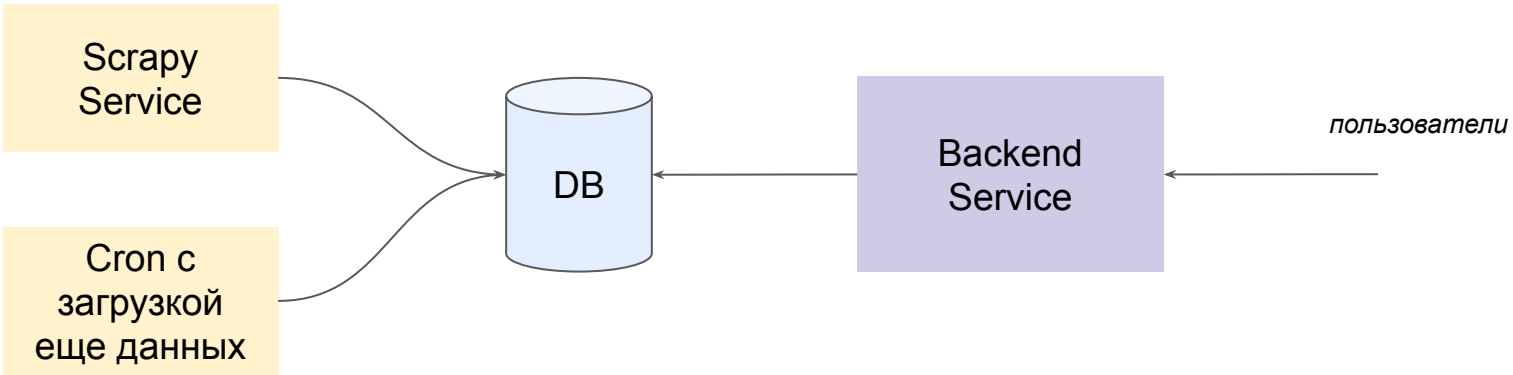


https://www.openstreetmap.org

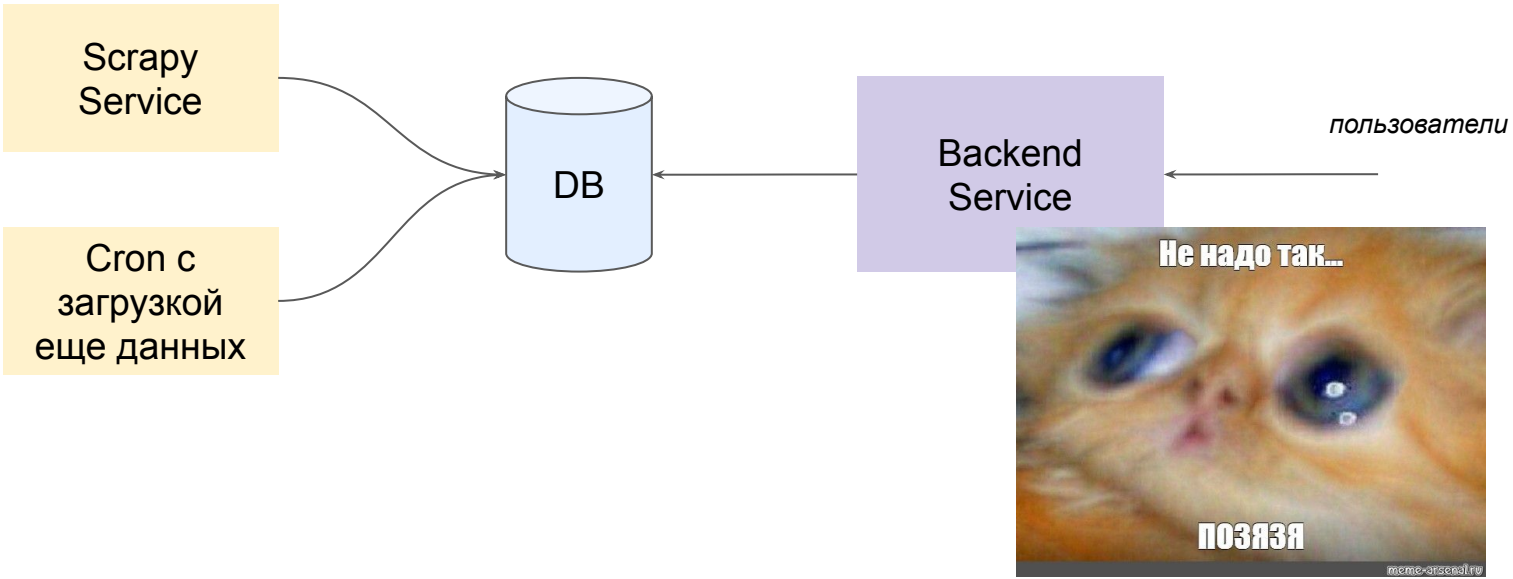
1. Собрать данные
2. Привести их в нужный формат
3. Сохранить к себе в базу



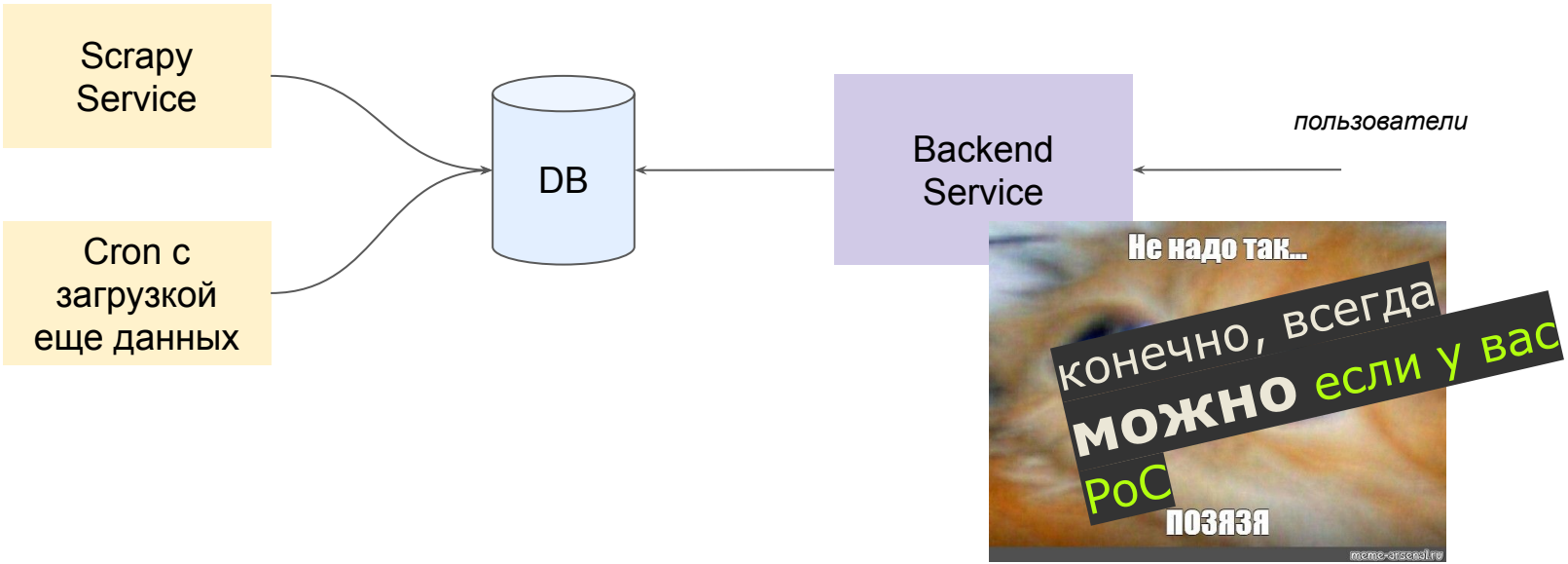
Scrapy … cron-job какой-нибудь … кладем все в основную базу в новые таблички - профит …



Scrapy … cron-job какой-нибудь … кладем все в основную базу в новые таблички - профит …



Scrapy … cron-job какой-нибудь … кладем все в основную базу в новые таблички - профит …



Дата-”особенности”

:: виды “особенностей”

1 Источники данных

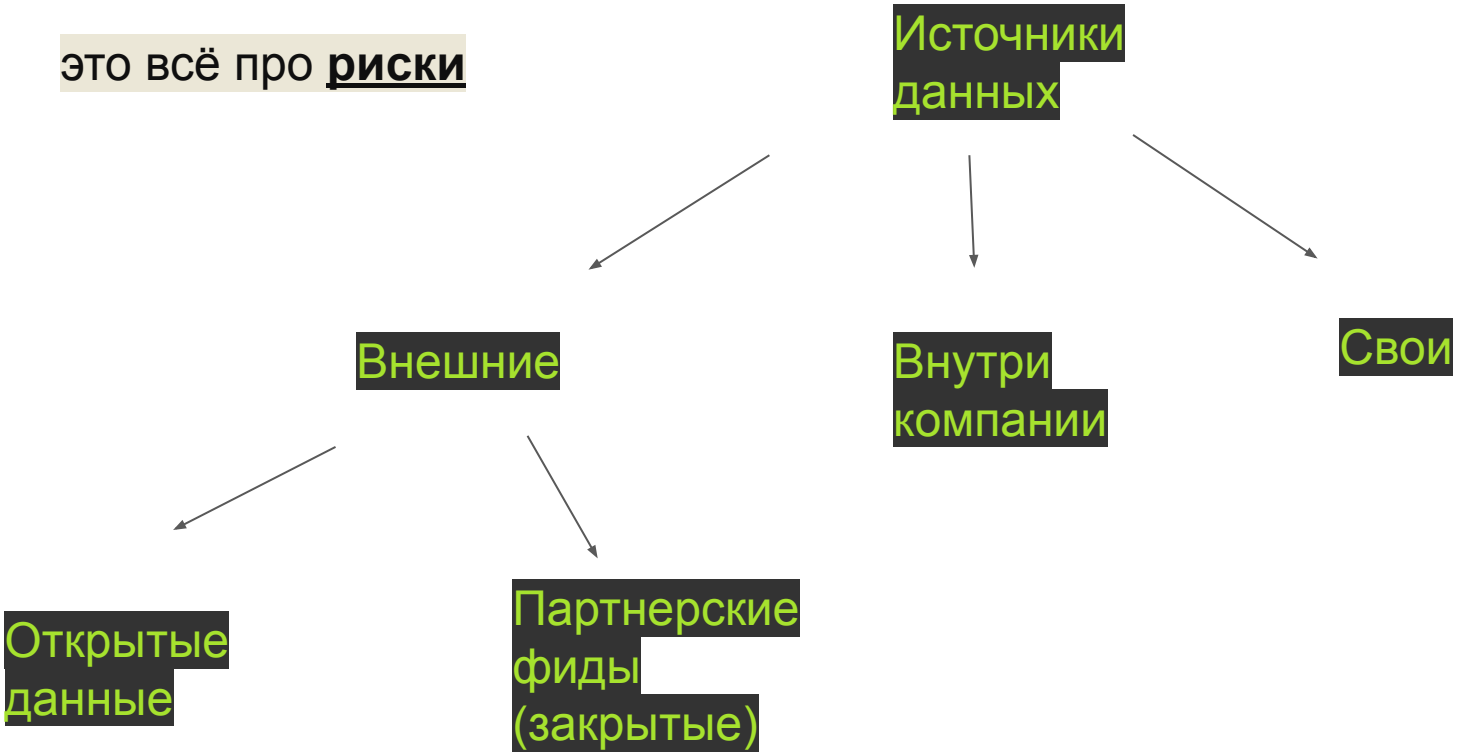
2 Работа со временем

3 Объем данных

4 Отношение к базам, хранилищам данных и SQL

5 Игнорирование Data-тулинга

это всё про **риски**



это всё про **риски**

это всё про **риски**

:: зачем помнить про контролируемость источников?

юзер/UI/client

- хороший
- monkey-тестер
- злоумышленник

:: авторы источников данных

- надежные
- мы заботимся о потребителях данных
- “не подумали” / нам откровенно все равно

* отсутствие данных за определенную дату
* некорректность в принципе
* частичная некорректность (ошибки)
* внедрение уязвимостей в данные
* изменение схемы данных / типов данных
* и тд

* отсутствие данных за определенную дату
* некорректность в принципе
* частичная некорректность (ошибки)
* внедрение уязвимостей в данные
* изменение схемы данных / типов данных
* и тд

:: и я даже не говорю про изначальное “качество” данных

```
SELECT distinct owner_name, owner_email, maintainer, maintainer_email FROM pypi.packages
LIMIT 100
```

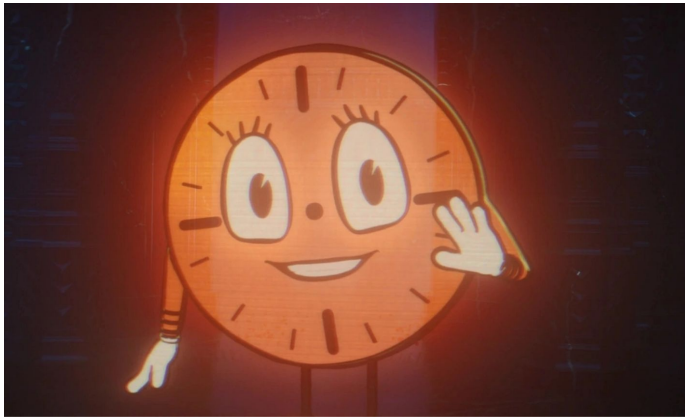
Data Output | Messages | Notifications

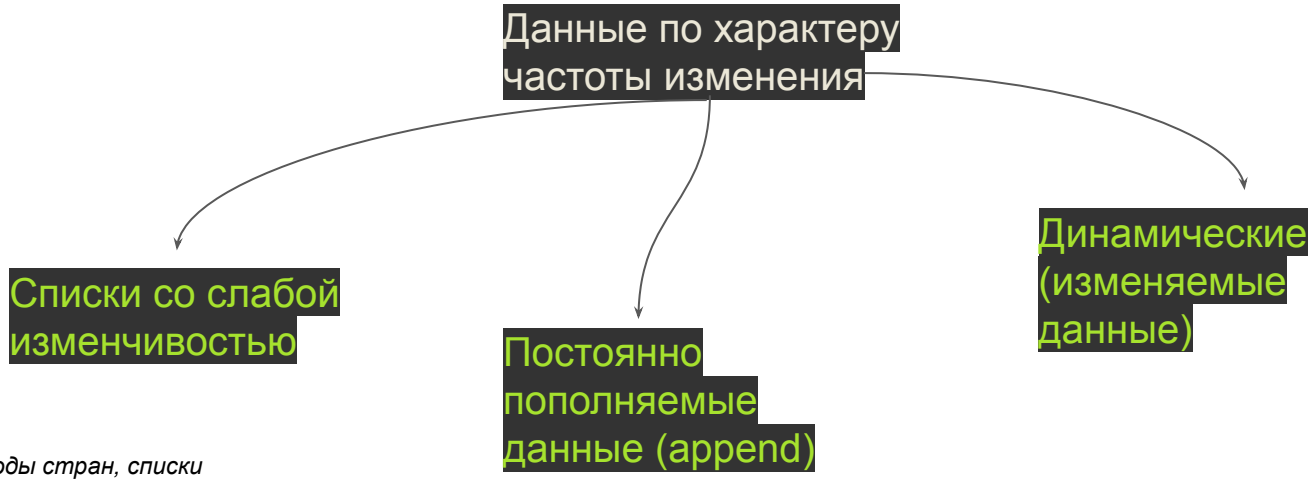
	owner_name text	owner_email text	maintainer text	maintainer_email text
20	Abdullah Diab	mpcabd@gmail.com	Abdullah Diab	mpcabd@gmail.com
21	OpenVoiceOS	builderjer@gmail.com		
22	Alexander Lyabah	looser@lyabah.com		
23		YL <fengyl@pku.edu.cn>		
24	Joshua McKiddy			
25	epsilon (nickN: kokomong)	kokomong1316@gmail.com		
26	William Ledda	villy80@hotmail.it		
27		loafthecomputerphile <loafdcomputerphile@gmail.com>		

74	Steven Harlow	stharlow@gmail.com
75	Tobias Pütz <puetztobias@gmail.c…	Tobias Pütz <puetztobias@gmail.com>, Daniël de Kok <me@danieldk.eu>
66	llc	UNKNOWN
67	Sina Sohrab	sinasohrab84@gmail.com

Режим по умолчанию:

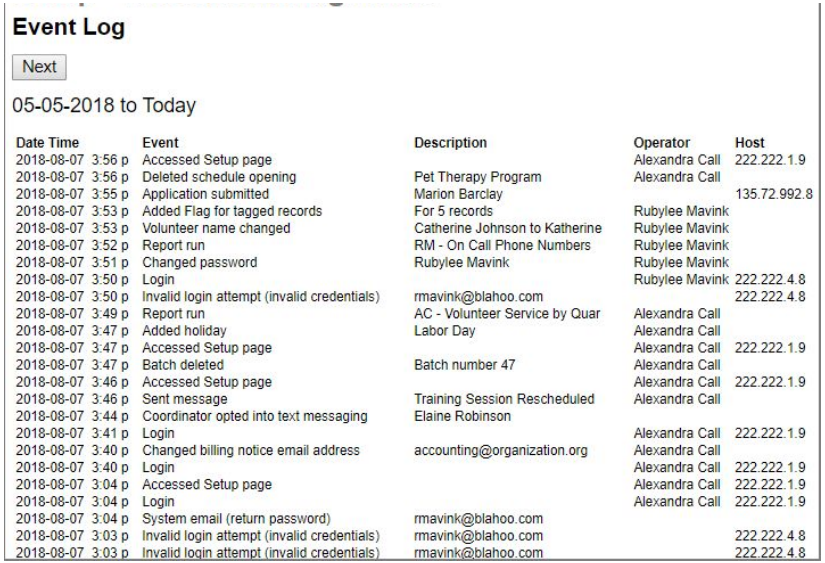
2 Работа со временем




Данные по характеру частоты изменения
Списки со слабой изменчивостью
Постоянно пополняемые данные (append)
Динамические (изменяемые данные)
коды стран, списки регионов в государствах и тд

Постоянно пополняемые данные (append)

новые данные дополняются, но предыдущие данные НЕ изменяются (списки погибших, списки исторических деятелей, списки исторических событий, список ЗАКРЫТЫХ сделок, лог событий и тд)



Event Log

Next

05-05-2018 to Today

Date Time	Event	Description	Operator	Host
2018-08-07 3:56 p	Accessed Setup page		Alexandra Call	222.222.1.9
2018-08-07 3:56 p	Deleted schedule opening	Pet Therapy Program	Alexandra Call	
2018-08-07 3:55 p	Application submitted	Marion Barclay		135.72.992.8
2018-08-07 3:53 p	Added Flag for tagged records	For 5 records	Rubylee Mavink	
2018-08-07 3:53 p	Volunteer name changed	Catherine Johnson to Katherine	Rubylee Mavink	
2018-08-07 3:52 p	Report run	RM - On Call Phone Numbers	Rubylee Mavink	
2018-08-07 3:51 p	Changed password	Rubylee Mavink	Rubylee Mavink	
2018-08-07 3:50 p	Login		Rubylee Mavink	222.222.4.8
2018-08-07 3:50 p	Invalid login attempt (invalid credentials)	rmavink@blahoo.com		222.222.4.8
2018-08-07 3:49 p	Report run	AC - Volunteer Service by Quar	Alexandra Call	
2018-08-07 3:47 p	Added holiday	Labor Day	Alexandra Call	
2018-08-07 3:47 p	Accessed Setup page		Alexandra Call	222.222.1.9
2018-08-07 3:47 p	Batch deleted	Batch number 47	Alexandra Call	
2018-08-07 3:46 p	Accessed Setup page		Alexandra Call	222.222.1.9
2018-08-07 3:46 p	Sent message	Training Session Rescheduled	Alexandra Call	
2018-08-07 3:44 p	Coordinator opted into text messaging	Elaine Robinson		
2018-08-07 3:41 p	Login		Alexandra Call	222.222.1.9
2018-08-07 3:40 p	Changed billing notice email address	accounting@organization.org	Alexandra Call	
2018-08-07 3:40 p	Login		Alexandra Call	222.222.1.9
2018-08-07 3:04 p	Accessed Setup page		Alexandra Call	222.222.1.9
2018-08-07 3:04 p	Login		Alexandra Call	222.222.1.9
2018-08-07 3:04 p	System email (return password)	rmavink@blahoo.com		
2018-08-07 3:03 p	Invalid login attempt (invalid credentials)	rmavink@blahoo.com		222.222.4.8
2018-08-07 3:03 p	Invalid login attempt (invalid credentials)	rmavink@blahoo.com		222.222.4.8

ИСТОЧНИК
https://www.volgistics.com/help/volgistics-platform/event-log-overview/

Динамические (изменяемые данные)

*Данные могут быть обновлены, дополнены, исправлены **в любой момент времени за любой момент времени***

Data Output | Messages | Notifications

	id [PK] text	modified timestamp with time zone	published timestamp with time zone	aliases text[]	summary text	affected jsonb
1	GHSA-76x4-hr82-cg3m	2024-02-16 09:40:05.813412+01	2022-05-24 18:47:43+02	{CVE-2019-10331}	Jenkins ElectricFlow Plugin cross-site reque...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
2	GHSA-4rgh-jx4f-qfcq	2023-11-08 05:03:35.243834+01	2022-05-24 19:37:16+02	{CVE-2020-35669}	http before 0.13.3 vulnerable to header inject...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
3	GHSA-4xh4-v2pq-jvhm	2023-11-08 05:11:39.890896+01	2022-09-20 00:47:29+02	{CVE-2023-22963}	personnummer/dart vulnerable to Improper I...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
4	GHSA-22c3-whjv-hrfm	2024-02-16 09:08:48.548868+01	2023-08-16 17:30:17+02	{CVE-2023-40337}	Jenkins Folders Plugin cross-site request for...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
5	OSV-2016-1	2022-04-13 05:04:30.875874+02	2021-01-13 22:56:22.388453+01	[null]	UNKNOWN READ in mprint	[{"ranges": [{"repo": "https://github.com/file/file.git", "type":
6	GHSA-vvfj-p4jf-j8rm	2024-02-16 09:18:22.972089+01	2022-05-24 18:44:55+02	{CVE-2019-10308}	Missing permission check in Jenkins Static ...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
7	OSV-2016-2	2022-04-13 05:04:30.860493+02	2021-01-13 22:57:40.577225+01	[null]	UNKNOWN READ in file_regexec	[{"ranges": [{"repo": "https://github.com/file/file.git", "type":
8	PYSEC-2023-114	2024-05-03 12:42:09.106815+02	2023-07-06 23:15:00+02	{CVE-2023-29824}		[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
9	PYSEC-2019-33	2023-03-14 08:01:09.341606+01	2019-06-27 16:15:00+02	[null]		[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
10	GHSA-9324-jv53-9cc8	2024-02-16 09:19:11.352648+01	2023-03-21 23:41:11+01	{CVE-2021-31402}	dio vulnerable to CRLF injection with HTTP ...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
11	DLA-3484-1	2024-01-09 21:33:30.378406+01	2023-07-08 02:00:00+02	[null]	firefox-esr - security update	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
12	GHSA-hm7f-rq7q-j9xp	2023-11-08 05:11:08.34192+01	2023-01-20 04:30:28+01	{CVE-2023-0410}	@builder.io/qwik vulnerable to Cross-site Scr...	[{"ranges": [{"type": "SEMVER", "events": [{"introduced": "0"},
13	GHSA-jrfm-2h82-xg28	2024-05-03 12:30:58.637089+02	2023-07-06 23:30:28+02	[null]	Withdrawn: Use after free in SciPy	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
14	GHSA-h56g-gq9v-vc8r	2024-02-16 09:10:09.649141+01	2023-12-05 19:15:02+01	{CVE-2023-49080}	jupyter-server errors include tracebacks with ...	[{"ranges": [{"type": "ECOSYSTEM", "events": [{"introduced":
15	OSV-2024-102	2024-02-15 01:00:47.674761+01	2024-02-15 01:00:47.674187+01	[null]	Heap-buffer-overflow in mrb_memsearch	[{"ranges": [{"repo": "https://github.com/mruby/mruby", "typ
16	OSV-2024-103	2024-02-15 01:00:51.271973+01	2024-02-15 01:00:51.271367+01	[null]	Heap-use-after-free in xmlRemoveID	[{"ranges": [{"repo": "https://gitlab.gnome.org/GNOME/libxn

* как мы поддерживаем актуальность данных (обнаруживаем и забираем обновления)

* а как часто мы это делаем?

* как мы храним данные (какой у них будет объем и как это объем будет расти во времени)

* как мы процессим данные (с какой частотой, какой размер одной итерации процессинга)

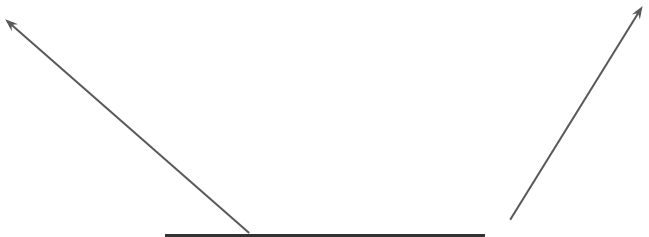
:: на что влияет изменчивость данных и частота их обновления

от 1 мин

Batch Processing

real-time

Streaming Processing



Это про то как
быстро мы хотим
видеть данные у себя

даты в данных

даты системные оркестрации (дата процессов)

когда мы это процессили/запускали?

даты сохранения в нашей базе

когда мы это сохраняли к себе?

2 ключевых процесса

оба включают трансформацию при необходимости

Забор данных

Первичный (взять всё, что уже есть)

Забрать обновления (since / iteration)

Ре-процессинг

как мне исправить косяки не блокируя пользователей конечного результата?

как это сделать наиболее эффективно?

:: отвечаем себе всегда на некоторые вопросы:

1) как мне понять, что было что-то изменено на источнике?

2) что я буду делать, если окажется, что данные были изменены на источнике - 1 колонка / добавлены N-строк и тд?

3) что я буду делать, если окажется, что изменения в моем коде были неверными?

4) что я буду делать, если мне надо будет добавить еще 1 колонку в мои данные?

*Помним про **“нюанс”***
контролируемости источников

Приходит клиент
~~в бар~~ с проблемой…

PYSEC-2024-60

Import Source	https://github.com/pypa/advisory-database/blob/main/vulns/idna/PYSEC-2024-60.y
JSON Data	https://api.osv.dev/v1/vulns/PYSEC-2024-60
Aliases	CVE-2024-3651 GHSA-jjg7-2v4v-x38h
Published	2024-07-07T18:15:00Z
Modified	2024-07-11T17:42:33.704488Z
Severity	7.5 (High) CVSS_V3 - CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H
Summary	[none]
Details	A vulnerability was identified in the kjd/idna library, specifi function's handling of crafted input strings, which can lead triggered by a crafted input that causes the `idna.encod`

Type	GIT	
Repo	https://github.com/kjd/idna	
Events	Introduced	0
	Fixed	1d365e17e10d72d0b7876316
Type	ECOSYSTEM	
Events	Introduced	0.1
	Fixed	3.7

Commit

✓ **PYSEC-2024-60: fix fixed field**

Signed-off-by: William Woodruff <william@trailofbits.com>

main (#187)

woodruffw committed on Jul 15 Verified

Showing **1 changed file** with **1 addition** and **1 deletion**.

2 vulns/idna/PYSEC-2024-60.yaml

```
@@ -24,7 +24,7 @@ affected:
24 24     - type: ECOSYSTEM
25 25       events:
26 26       - introduced: "0.1"
27    -     fixed: "3.7"
   27 +     - fixed: "3.7"
28 28     versions:
29 29     - "0.1"
30 30     - "0.2"
```

действительно, зачем менять дату обновления…

в зависимости от предметной области / нужных трансформаций и используемого тулинга

что-то на Spark-овском

```
baseDir = os.path.join('databricks-datasets')
inputPath = os.path.join('cs100', 'lab1', 'data-001', 'shakespeare.txt')
fileName = os.path.join(baseDir, inputPath)

shakespeareRDD = (sc.textFile(fileName, 8).map(removePunctuation))
                    .join(shakespeareRDD
                       .zipWithIndex()
                       .map(lambda (l, num): '{0}: {1}'.format(num, l))
                       .take(15))
```

https://juheck.gitbooks.io/demo-spark/content/example-wordcount.html

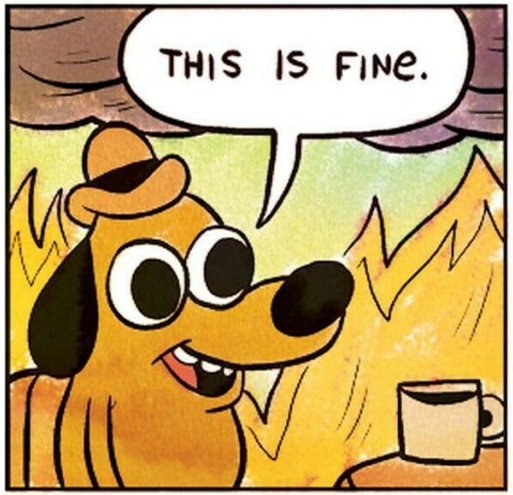
```python
def load_recipes_since(self, since: datetime) -> int:
    github_paths = ConanGithubUpdater().get_the_changed_components_since_date(since)
    # мы из путей в виде 'recipes/s2n/all/conandata.yml' хотим получить только имена пакетов - s2n
    recipes_names = set([path.split('/', 2)[1] for path in github_paths])  # noqa C403
    recipes = ConanAPIHelper().get_recipes_by_names(recipes_names)
    return self.serialize_and_save_recipes(recipes)

async def get_errors_packages_names(self) -> int:
    self.error_recipes_names = [f'{package.name}/{package.version}'
                                async for package in self.get_errors_packages()]

def load_only_errors_recipes(self) -> int:
    asyncio.run(self.get_errors_packages_names)
    recipes = ConanAPIHelper().get_recipes_by_names(self.error_recipes_names)
    return self.serialize_and_save_recipes(recipes)
```

а еще может быть просто чистый Python (вот у нас именно так)

все те же проблемы - баги,
неполные тест-кейсы,
python-relative проблемы, а
еще взять интеграции с jvm,
криворукость …



3 Про объемы данных

“Большая таблица”

5 колонок

BigAFTable

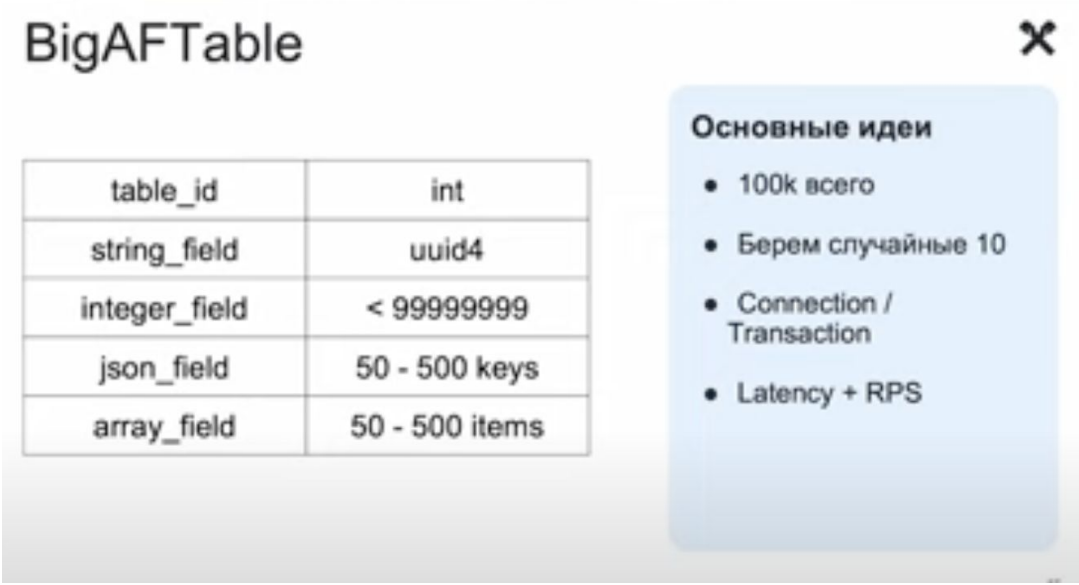
table_id	int
string_field	uuid4
integer_field	< 99999999
json_field	50 - 500 keys
array_field	50 - 500 items

Основные идеи

- 100k всего
- Берем случайные 10
- Connection / Transaction
- Latency + RPS

Слайд и цитата из доклада Pycon 2024 “Михаил Гурбанов. На старт. Внимание. RUST”

Это
МАЛЕНЬКАЯ
ТАБЛИЦА
даже для
PostgreSQL



Слайд и цитата из доклада Pycon 2024 “Михаил Гурбанов. На старт. Внимание. RUST”

:: приблизительные границы

Маленький объем данных

Таблицы
< 1 млн записей

Общий объем процессинга < 500 Мб / сутки

прирост < 6 Гб / мес

“Средние” данные

Таблицы
< 100 (?) млн записей (можно и больше, но тогда уже играем в партиции)

Общий объем процессинга < ? Гб / сутки

Big Data

Все что > ? млн записей

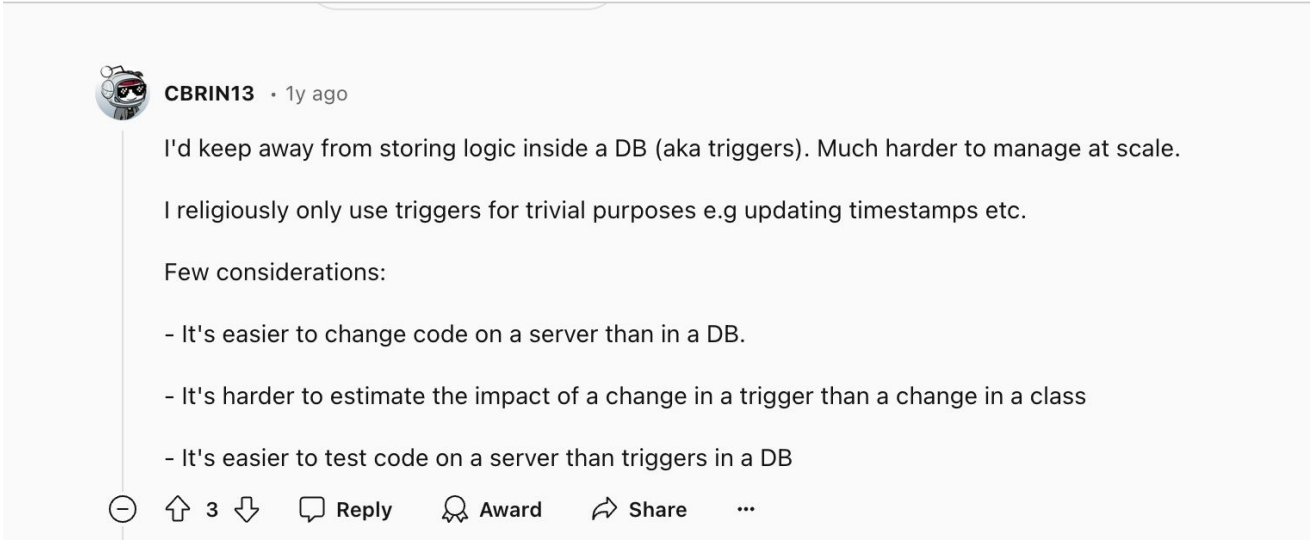
Общий объем процессинга > 100 Гб / сутки

тут может быть и по 5 Пт / сутки и больше

4 Отношение к базам, хранилищам данных, и SQL

triggers

тут должны быть мемы про “как меня видят бэкендеры / как меня видят DBA”



CBRIN13 • 1y ago

I'd keep away from storing logic inside a DB (aka triggers). Much harder to manage at scale.

I religiously only use triggers for trivial purposes e.g updating timestamps etc.

Few considerations:

- It's easier to change code on a server than in a DB.

- It's harder to estimate the impact of a change in a trigger than a change in a class

- It's easier to test code on a server than triggers in a DB

3 Reply Award Share ...

https://www.reddit.com/r/webdev/comments/158g26a/backend_design_database_triggers_vs_serverside/

triggers

реальность такова, что если вы можете заменить триггер в базе данных на код в Python - вам пока что реально не нужен был триггер

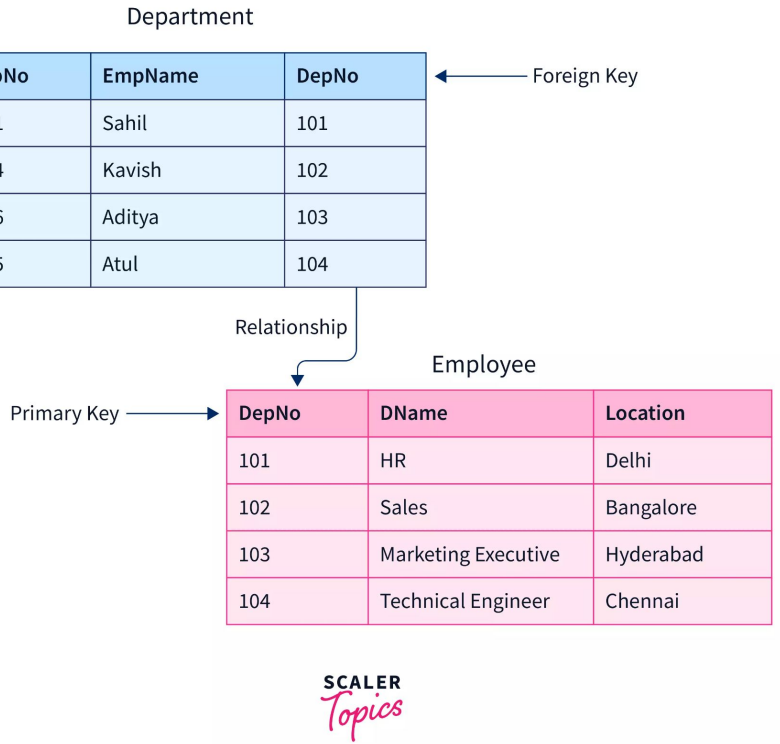
If those triggers affect database integrity or data integrity (checking things, inserting other records, making backup copies of records before modification or maybe something else) then it should be database triggers, because you never know how many of your frontends will operate on the database at a given time and syncronizing them by yourself is a huge pain in the ... and it's also prone to errors while you already have a less-painful and less error-prone method at hand.

https://stackoverflow.com/questions/18755838/which-is-more-efficient-making-database-triggers-or-handling-all-trigger-events

triggers - use case

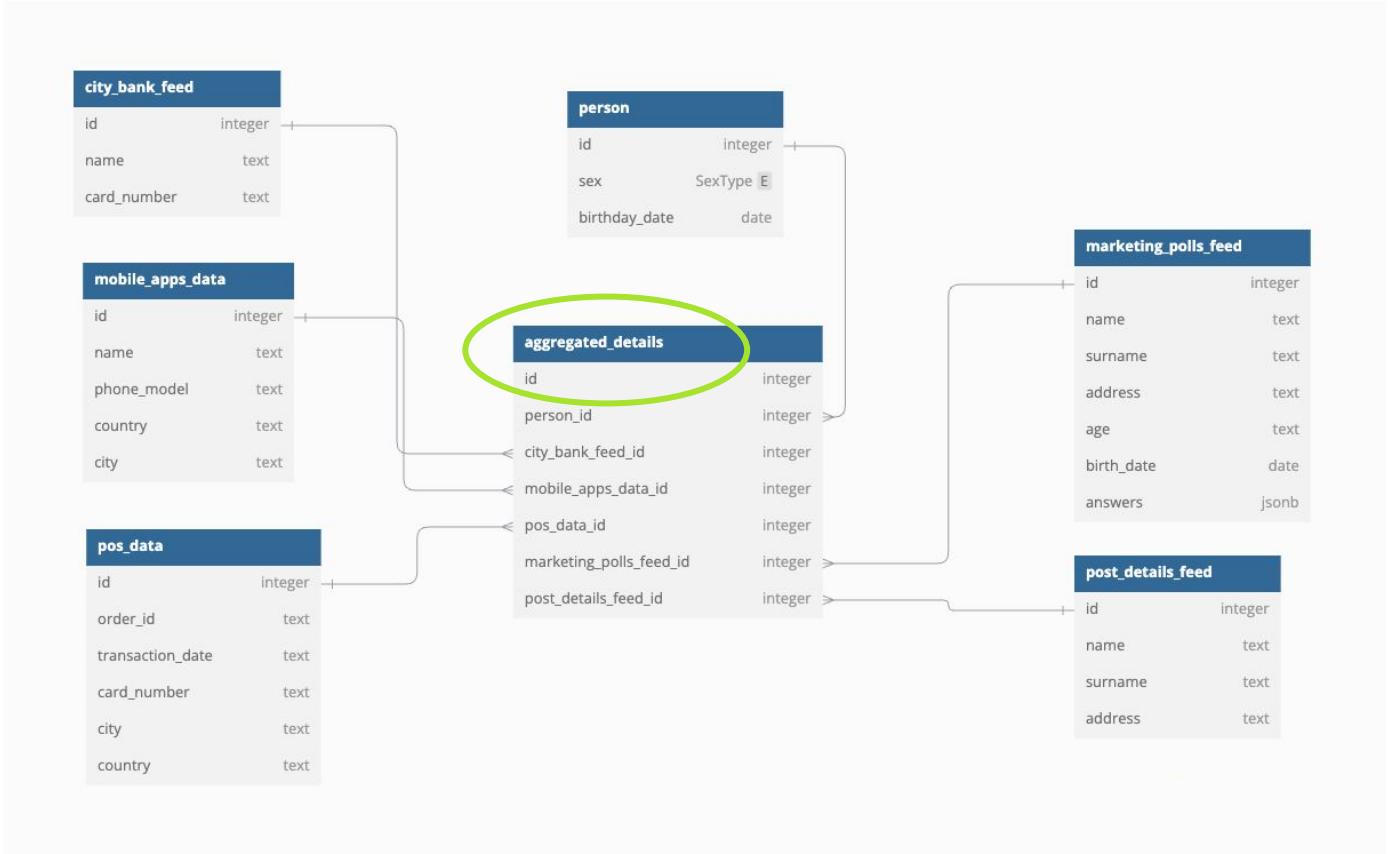
	purl text	vuln_id text	source text	fixed_version text	operation_type text	operation_timestamp timestamp with time zone
1	pkg:npm/%40illimity/components@3.31.0	MAL-2024-7860	osv	[null]	insert	2024-08-04 04:32:48.7568
2	pkg:deb/debian/libssl-dev@3.3.1-2?arch=s390x&distro=sid&upstream=openssl	CVE-2007-6755	debian	[null]	insert	2024-08-04 05:13:10.6371
3	pkg:deb/debian/libssl-dev@3.3.1-2?arch=s390x&distro=sid&upstream=openssl	CVE-2010-0928	debian	[null]	insert	2024-08-04 05:13:10.6371
4	pkg:deb/debian/libssl-dev@3.3.1-2?arch=s390x&distro=sid&upstream=openssl	CVE-2024-5535	debian	[null]	insert	2024-08-04 05:13:10.6371
5	pkg:deb/debian/gir1.2-glib-2.0-dev@2.81.1-3?arch=riscv64&distro=sid&upstream=glib2.0	CVE-2012-0039	debian	[null]	insert	2024-08-04 05:12:43.2646
6	pkg:deb/debian/tin@1:2.6.4~20240801-1?arch=arm64&distro=sid	CVE-2017-17520	debian	[null]	insert	2024-08-04 05:12:44.1051
7	pkg:pypi/ansible@2.9.22	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
8	pkg:pypi/ansible@2.8.5	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
9	pkg:pypi/ansible@2.9.25rc1	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
10	pkg:pypi/ansible@2.9.16rc1	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
11	pkg:pypi/ansible@2.7.18	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
12	pkg:pypi/ansible@2.8.4	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
13	pkg:pypi/ansible@2.6.13	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
14	pkg:pypi/ansible@8.4	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
15	pkg:pypi/ansible@9a3	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
16	pkg:pypi/ansible@2.9.26	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357
17	pkg:pypi/ansible@2.10.7	PYSEC-2021-125	osv	[null]	delete	2024-08-05 21:01:31.7357

foreign keys



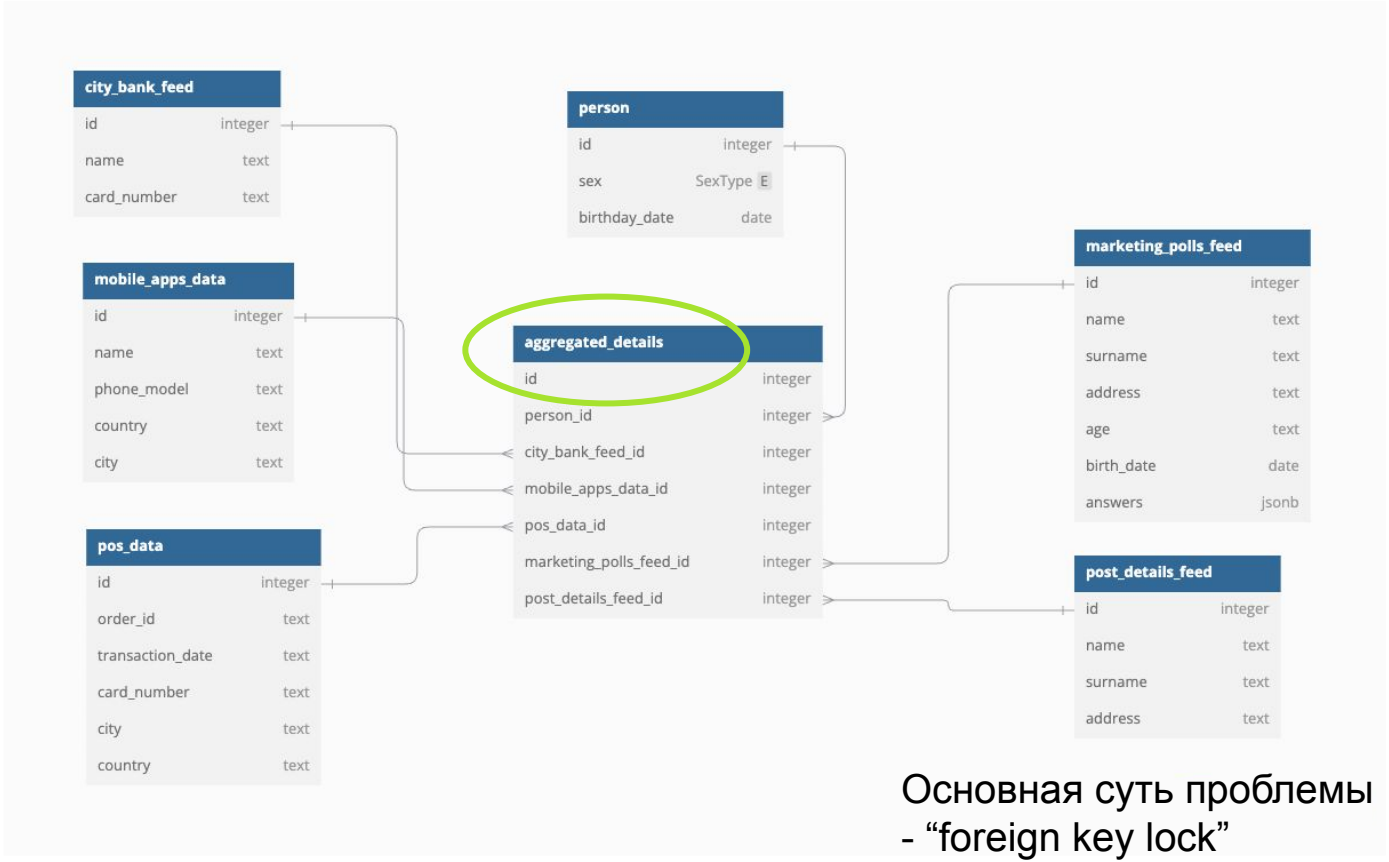
картинка с https://www.scaler.com/topics/sql/foreign-key-in-sql/

foreign keys



foreign keys

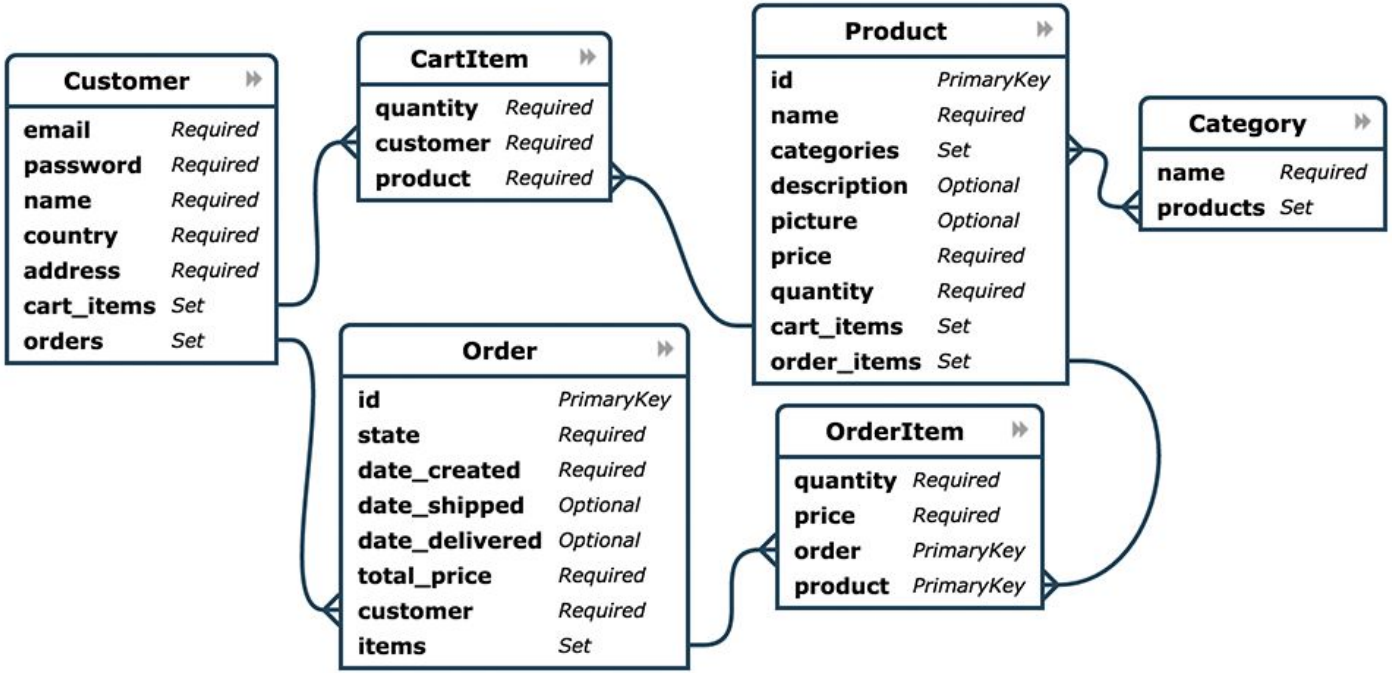
представьте, если вставка в каждый фид идет постоянно non-stop, а таблицу еще кто-то читает в этот момент?



Основная суть проблемы - “foreign key lock”

ORM

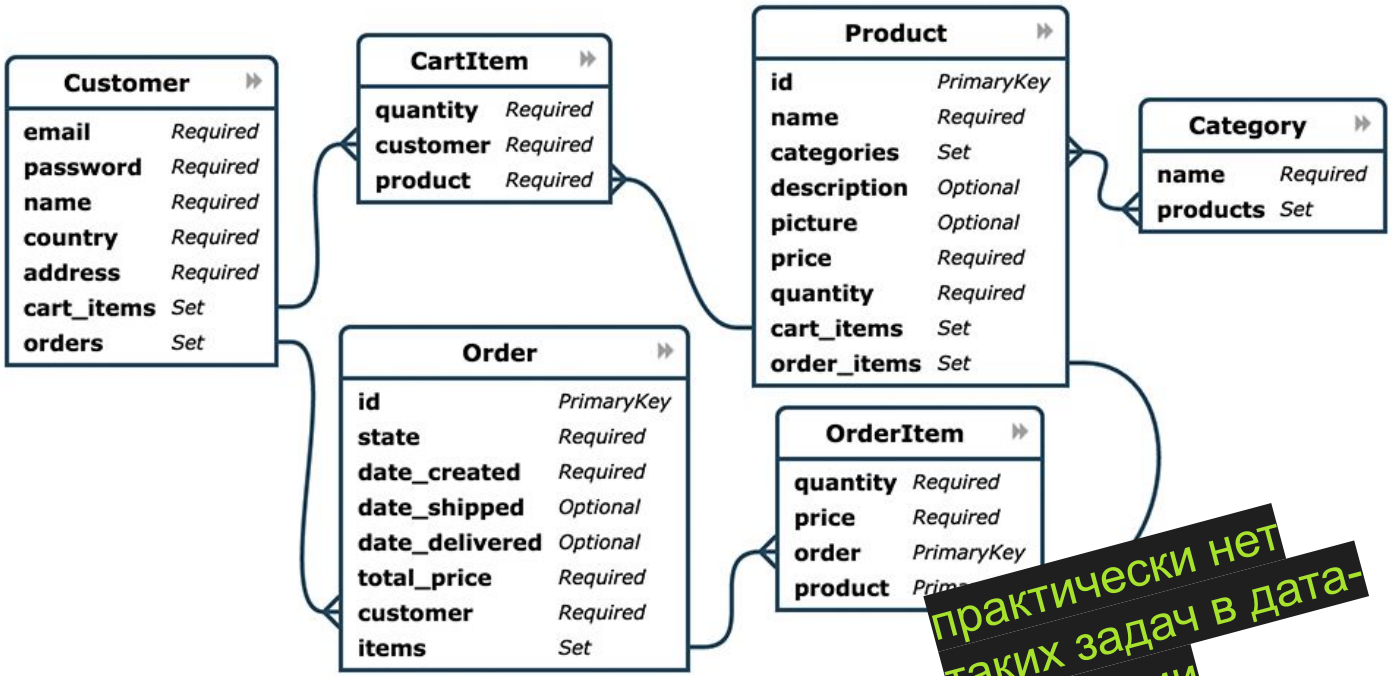
Object–**relational** mapping



https://ponyorm.org/

ORM

Object–**relational** mapping



https://ponyorm.org/

ORM

ORM != работа с базой “удобно” из кода

ORM НЕ УДОБНЕЕ SQL

Расскажите как сделать

INSERT INTO table_name SELECT … FROM

Или может TEMPORARY таблицу?

https://ponyorm.org/

SQL-features

```
select
        message_id,
        package_manager,
        match_type,
        JSONExtractKeysAndValuesRaw(nodes).2 as nodes_values,
        JSONLength(nodes) - 1 as nodes_count,
        arrayMap( x -> arrayCompact(arrayMap(i -> JSONExtractString(i, 'fixed_version'), x)), arrayMap(
                x -> JSONExtractArrayRaw(x),
                        arrayFilter( x -> x not ilike '[]', arrayMap(
                x -> JSONExtractString(x, 'vulnerabilities'),
                JSONExtractKeysAndValuesRaw(nodes).2))))
         as deps_vulns_from_graph,
        arrayCompact(arrayMap(x-> JSONExtractString(x, 'name'),
        JSONExtractArrayRaw(JSONExtractString(JSONExtractString(nodes, 'root'), 'dependencies')))) as
root_deps,
        case when isValidJSON(name) then JSONExtractString(name, 'timestamp') else null end as timestamp,
        case when isValidJSON(name) then JSONExtractString(name, 'owner') else owner end as owner,
        case when isValidJSON(name) then JSONExtractString(name, 'version') else null end as cs_version,
        name,
        version,
        manifest,
        lockfile,
        extra,
        is_resolved,
        started_at,
        ended_at,
        result,
        duration
from main_data
```

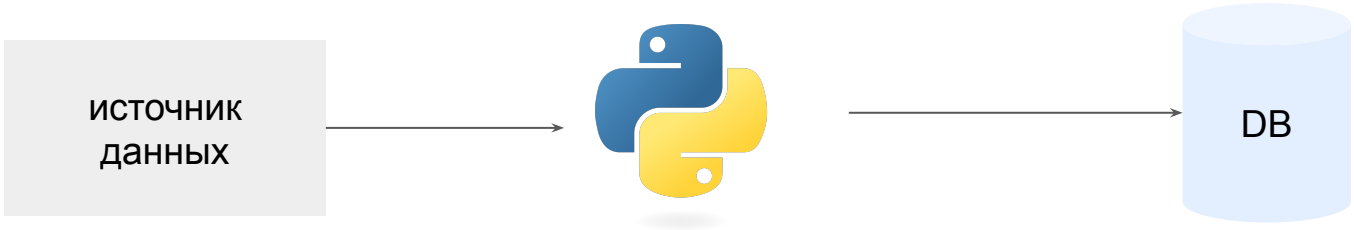
в этой квере было где-то 200 строк, ну и не то что бы это большая sql…

SQL-features

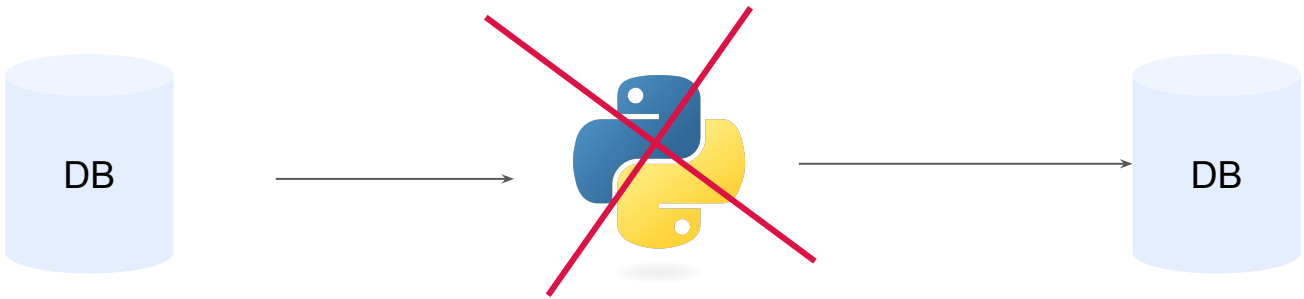
- CREATE TEMPORARY TABLE
- CREATE MATERIALIZED VIEWS
- Table Partitioning
- функции работы со строками https://clickhouse.com/docs/en/sql-reference/functions/string-functions (это как пример) , массивами, json-ами - всем чем только можно (Трансформация данных в базе)
- и тд

Почему так?

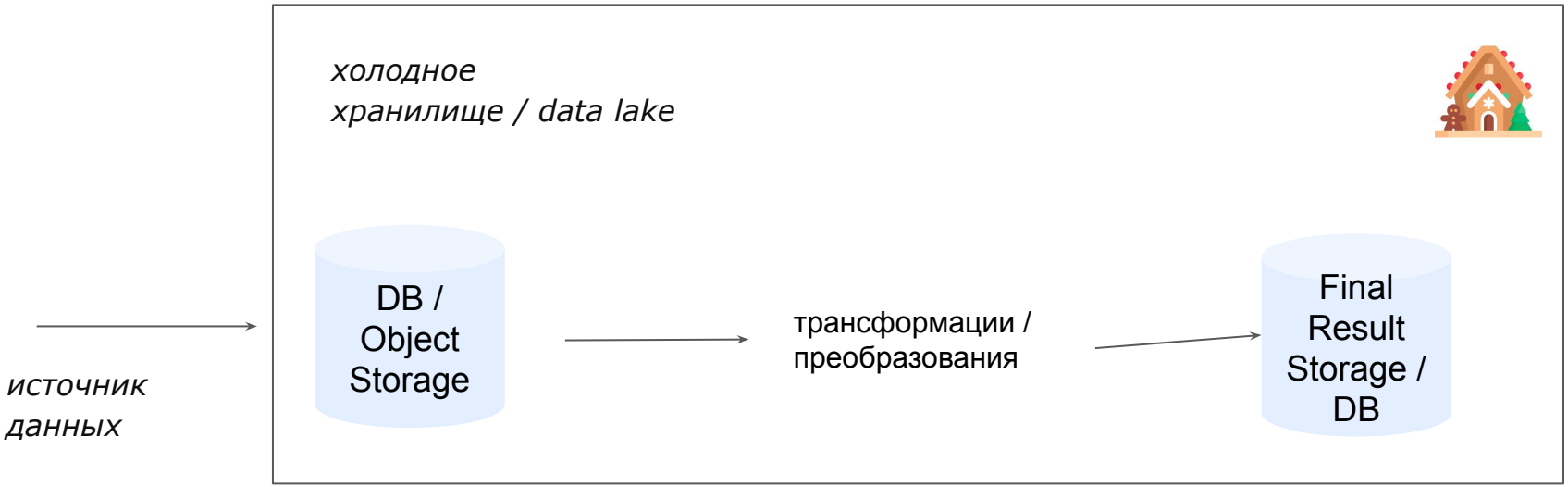
а давайте теперь просто задумаемся про косты на networking, perfomance за счет сериализаци/десериализации в Python

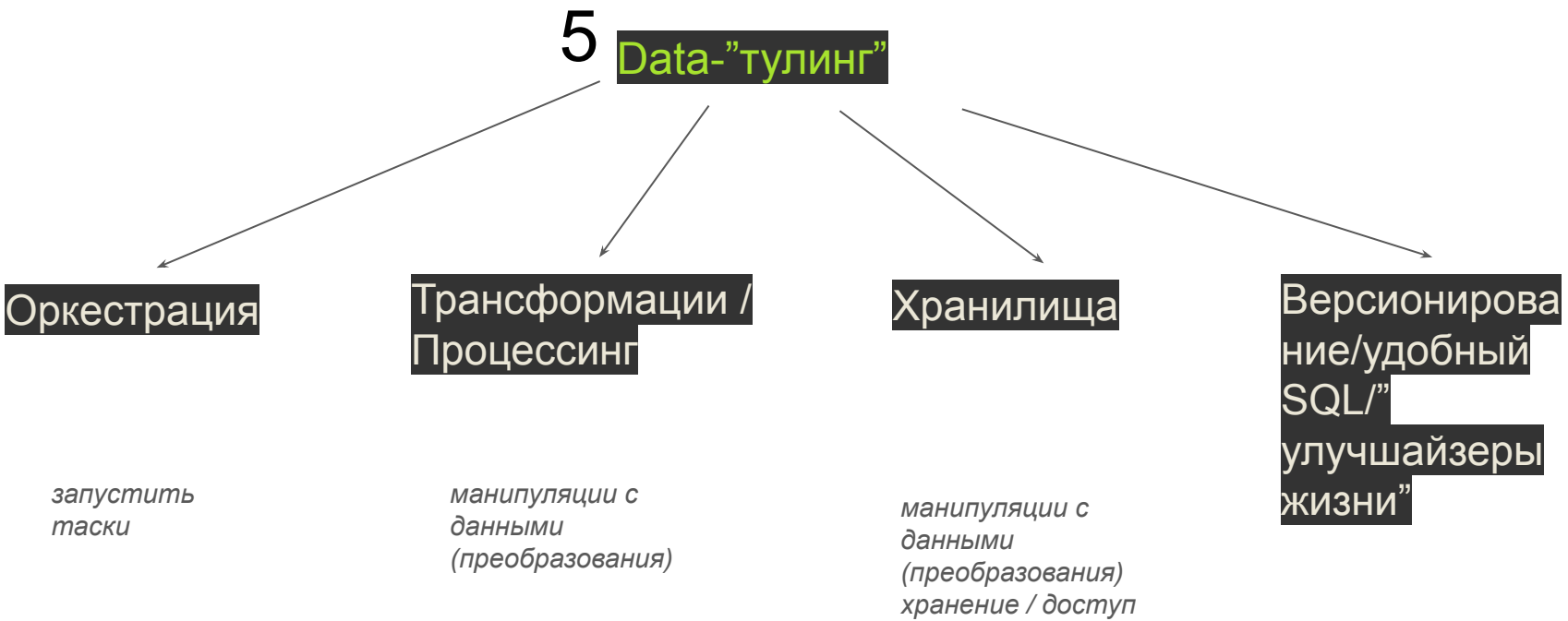


если мы в ситуации, когда данные уже в нашей базе



:: поэтому


холодное
хранилище / data lake
DB /
Object
Storage
трансформации /
преобразования
Final
Result
Storage /
DB
источник
данных


5
Data-”тулинг”
Оркестрация
Трансформации / Процессинг
Хранилища
Версионирование/удобный SQL/” улучшайзеры жизни”
запустить таски
манипуляции с данными (преобразования)
манипуляции с данными (преобразования) хранение / доступ

- помним про **хвосты ре-процессинга** в зависимости от объема данных
- на этапе планирования / декомпозиции задачи помним о первом пункте
- в 80% случае Data-задача сделана ни тогда, когда у вас код готов, а когда валидные данные оказались в Базе

- мест, **где все может пойти не так** гораздо больше чем в backend-ах
- зато при нормальной архитектуре (которую очень просто реализовать), это не должно вести ни к каким PROD-проблемам
- занимайтесь дата-инженерией, у нас весело
- думайте про размеры данных, как часто их забирать, по каким триггерам (базируясь на чем?), выбирайте правильные инструменты

Спасибо,
у меня всё!

Вопросы?

(накидать помидоров
можно в отзывы)

меня всегда можно найти тут > https://github.com/**xnuinside**