

Event Sourcing 101

@fahchen 2020.12.11

面向数据库编程

面向数据库编程

由应用维护数据的当前状态，在用户使用时对数据进行更新

- 数据如何存储
- 数据如何流动

面向数据库编程

由应用维护数据的当前状态，在用户使用时对数据进行更新

- 从存储（数据库）中读取数据
- 根据业务逻辑（对数据）做出修改
- 更新数据的当前状态（通常会使用 transactions 去锁住对应的数据来防止冲突）

举个🍑

- user-1 存入 100 元
- user-2 存入 200 元
- user-3 存入 300 元

User	Balance
user-1	100
user-2	200
user-3	300

举个🍑

- user-1 存入 100 元
- user-2 存入 200 元
- user-3 存入 300 元

User	Balance
user-1	100
user-2	200
user-3	300

举个🍑

- user-1 存入 100 元
- user-2 存入 200 元
- user-3 存入 300 元
- user-2 取出 100 元

User	Balance
user-1	100
user-2	100
user-3	300




举个🍑

- user-1 存入 100 元
- user-2 存入 200 元
- user-3 存入 300 元
- user-2 取出 100 元
- user-3 向 user-1 转账 100 元

User	Balance
user-1	200
user-2	100
user-3	200

举个🍑

- user-3 取出 200 元？（防止双重花费）
- user-2 向 user-1 转 100 元？（防止透支）
- user-1 取出 100 元？（需要等待锁释放）

User	Balance
 user-1	200
 user-2	100
 user-3	200

Create, Read, Update, Delete

面向数据库编程

CRUD 的局限

- CRUD系统在数据库上直接执行更新操作，由于需要开销，会降低系统的性能与响应能力，限制可扩展性。
- 在一个由多个用户并发操作的领域中，数据更新更有可能起冲突，因为更新操作发生在同一条单独的数据项上。
- 除非在某个单独的日志中存在额外的审计机制来记录每个操作的细节，否则历史记录会丢失。

什么是 Event Sourcing ?

使用只可追加的存储来记录对数据所进行的所有操作，而不是存储领域数据的当前状态。

Event Sourcing Pattern

举个🍑

- user-1 存入 100 元
- user-2 存入 200 元
- user-3 存入 300 元
- user-2 取出 100 元
- user-3 向 user-1 转账 100 元

Event ID	Event
1	user-1 存入了 100 元
2	user-2 存入了 200 元
3	user-3 存入了 300 元
4	user-2 取出了 100 元
5	user-3 取出了 100 元
6	user-1 存入了 100 元

举个🍑

当前余额

ID	Event
1	user-1 存入了 100 元
2	user-2 存入了 200 元
3	user-3 存入了 300 元
4	user-2 取出了 100 元
5	user-3 取出了 100 元
6	user-1 存入了 100 元

User	Balance
user-1	200
user-2	100
user-3	200

只要事件足够多，足够细就越能还原当时的
所有的细节，就像拍电影一样。

"We may regard the present state of the universe as the effect of its past and the cause of its future. An intellect which at a certain moment would know all forces that set nature in motion, and all positions of all items of which nature is composed, if this intellect were also vast enough to submit these data to analysis, it would embrace in a single formula the movements of the greatest bodies of the universe and those of the tiniest atom; for such an intellect nothing would be uncertain and the future just like the past would be present before its eyes."

"我们可以把宇宙现在的状态看作是它历史的果，和未来的因。如果存在这么一个智慧，它在某一时刻，能够获知驱动这个自然运动的所有的力，以及组成这个自然的所有物体的位置，并且这个智慧足够强大，可以把这些数据进行分析，那么宇宙之中从最宏大的天体到最渺小的原子都将包含在一个运动方程之中；对这个智慧而言，未来将无一不确定，恰如历史一样，在它眼前一览无遗"

举个🍑

对 GDP 的贡献

Event ID	Event
1	user-1 存入了 100 元
2	user-2 存入了 200 元
3	user-3 存入了 300 元
4	user-2 取出了 100 元
5	user-3 取出了 100 元
6	user-1 存入了 100 元

User	GDP
user-1	200
user-2	300
user-3	400

In Memory

GIS

Searching

K/V Store

Events

Indexing

ElasticSearch

Caching

RDBMS

Graph Database



12.6晚打车至
解放路赫本酒吧

12.6打车至华林三路
海雾里酒馆

12.5后至下西顺城街
AMUSING CLUB酒吧

12.2-12.4在郫都区
坐地铁到达云景台

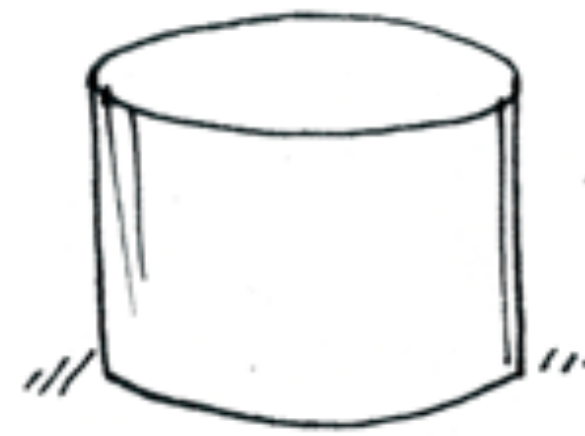
12.7凌晨打车
回到云景台

12.5乘坐公交、地到达
春熙路美甲店、餐馆、
中影和悦影院

12.6打车至
play house酒吧

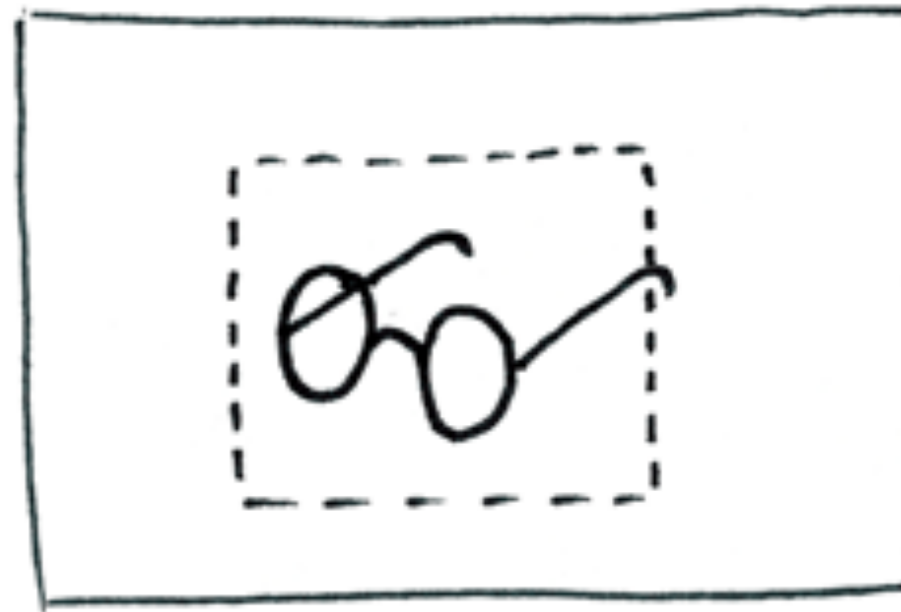
Command Query Responsibility Segregation

Denormalized read store
Subscribes to events
on the Write side



Query

Read side



Query response

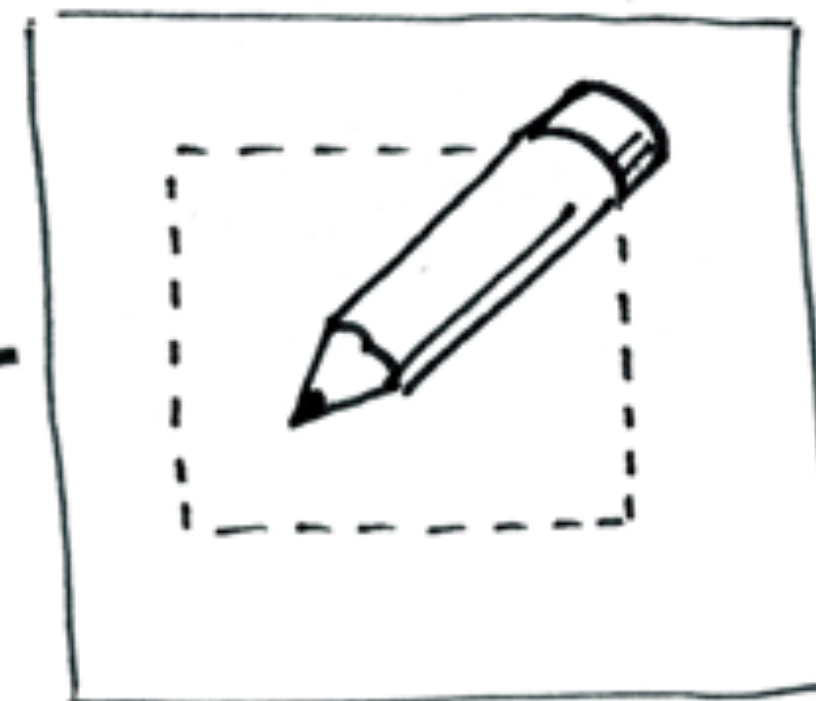


User views data
in the UI



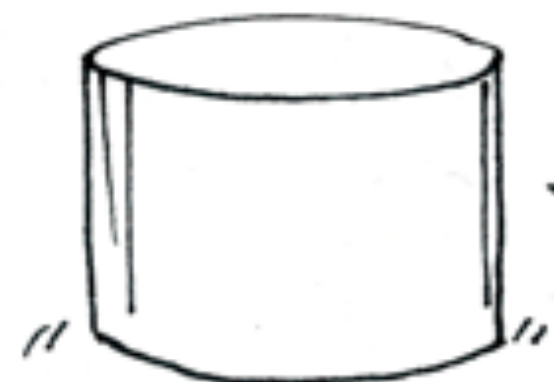
User makes a change
in the UI

Write side



Command

Append
events



Event store
Publishes events after
they have been saved

Events



UserBoundary 的维护



```
Airbase.IdentityAccess.Events.DepartmentCreated  
Airbase.IdentityAccess.Events.DepartmentUpdated  
Airbase.IdentityAccess.Events.DepartmentDiscarded
```

```
Airbase.IdentityAccess.Events.UserRegistered  
Airbase.IdentityAccess.Events.UserUpdated  
Airbase.IdentityAccess.Events.UserImported  
Airbase.IdentityAccess.Events.UserReceived  
Airbase.IdentityAccess.Events.UserRepelled
```

```
%UserBoundary{  
    user_uuids: [Ecto.UUID],  
    simple_department_uuids: [Ecto.UUID],  
    penetrating_department_uuids: [Ecto.UUID]  
}
```

```
%UserBoundary{  
    user_uuids: ['user-1'],  
    simple_department_uuids: ['department-1'],  
    penetrating_department_uuids: ['department-2']  
} =  
    ['user-1']  
++ users in 'department-1'  
++ users in 'department-2' and its descendents.
```

Benefits

- Scalable
 - Append only events (immutable)
 - Fits distributed (k/v) stores
 - Low-latency writes
 - Allows asynchronous processing
- History or audit
- Rebuild state at point in time
 - for indexing
 - for searching

Black side

- Event version and upgrade
- Slow read-side
- read-side dependency
- Eventual Consistency
- Immutable Events
- 不是所有的项目都适合 Event Sourcing!!!

More

- Domain Driven Design
- Data Consistency

<https://adventofcode.com/2020/>

参考链接

- <https://www.slideshare.net/LorenzoNicora/a-visual-introduction-to-event-sourcing-and-cqrs>
- <https://martinfowler.com/eaDev/EventSourcing.html>
- <https://docs.microsoft.com/en-us/azure/architecture/patterns/event-sourcing>