

07/10/2023

Version 2

SPRING CLOUD

JÉRÉMY PERROUULT

A decorative wavy line in light blue and white, flowing from the top left towards the bottom left of the slide.

DONNÉES

BASE DE DONNÉES PAR SERVICE

BDD PAR SERVICE

Dans le service **Commentaire**, une décision est prise

> La note est en fait une moyenne d'autres notes

- * Qualité du produit (0 à 5)
- * Rapport qualité-prix (0 à 5)
- * Facilité d'utilisation (0 à 5)

Implémenter ce changement

BDD PAR SERVICE

- La base de données est partagée entre les différents services
 - Ce qui induit un couplage fort (exécution & infrastructure) et peut occasionner des défaillances
 - En cas d'évolutions
 - En cas de surcharge des accès (accès concurrents par exemple)

NOTE On se privera de la gestion d'une transaction et des contraintes d'intégrité référentielles.
Il faudra réimplémenter ces mécanismes.

BDD PAR SERVICE

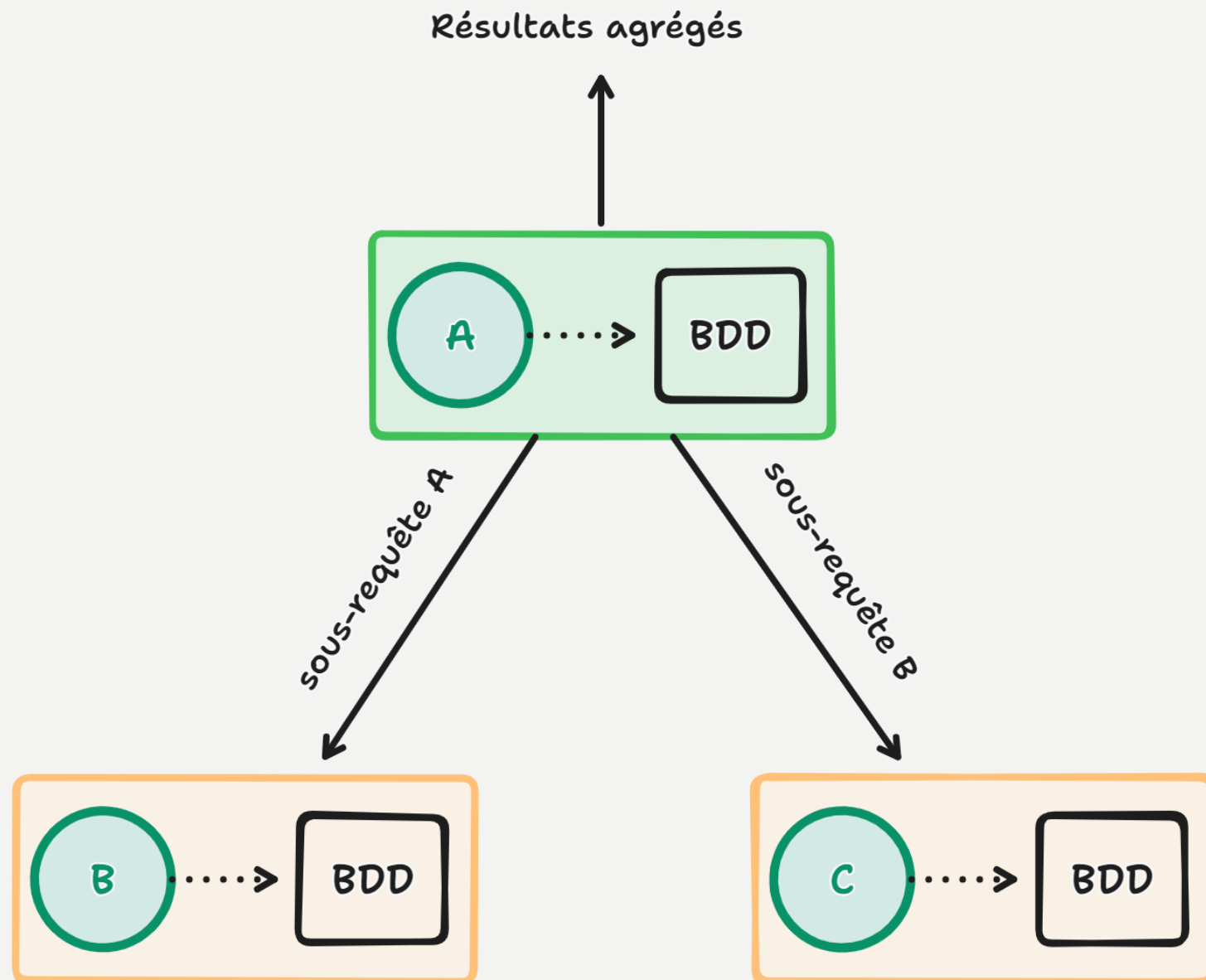
- Quand on évoque une « base de données » par service, il peut s'agir
 - D'autant de serveurs de données que de services : chaque service possède son serveur
 - D'un seul serveur de données « partagés » entre les services
 - Avec des tables « privées » : chaque service possède son lot de tables, les autres lui sont inaccessibles
 - Dans ce cas, on garde les contraintes d'intégrité référentielles, puisqu'il n'y a qu'un seul schéma
 - Avec des schémas privés : chaque service possède son schéma de données
 - *Dans ces 2 cas de figures, il est important d'avoir des utilisateurs avec des rôles bien définis pour accéder aux données*
 - D'un mix entre les 2 solutions précédentes

BDD PAR SERVICE

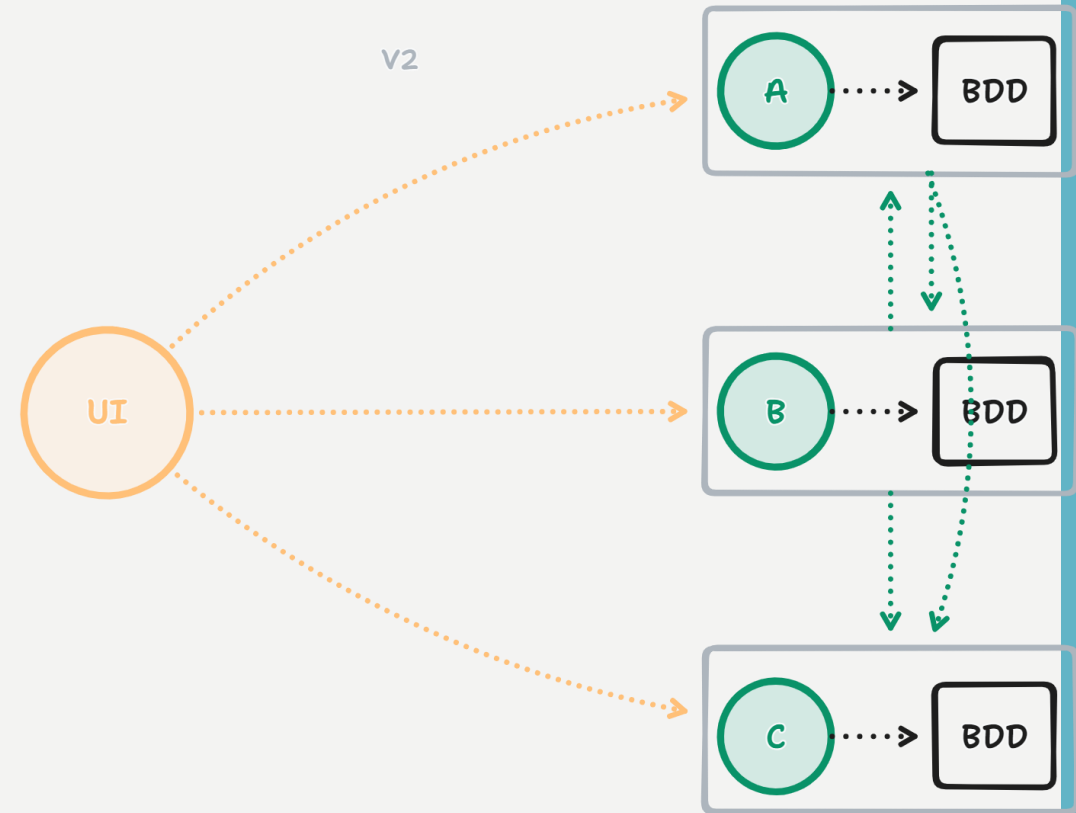
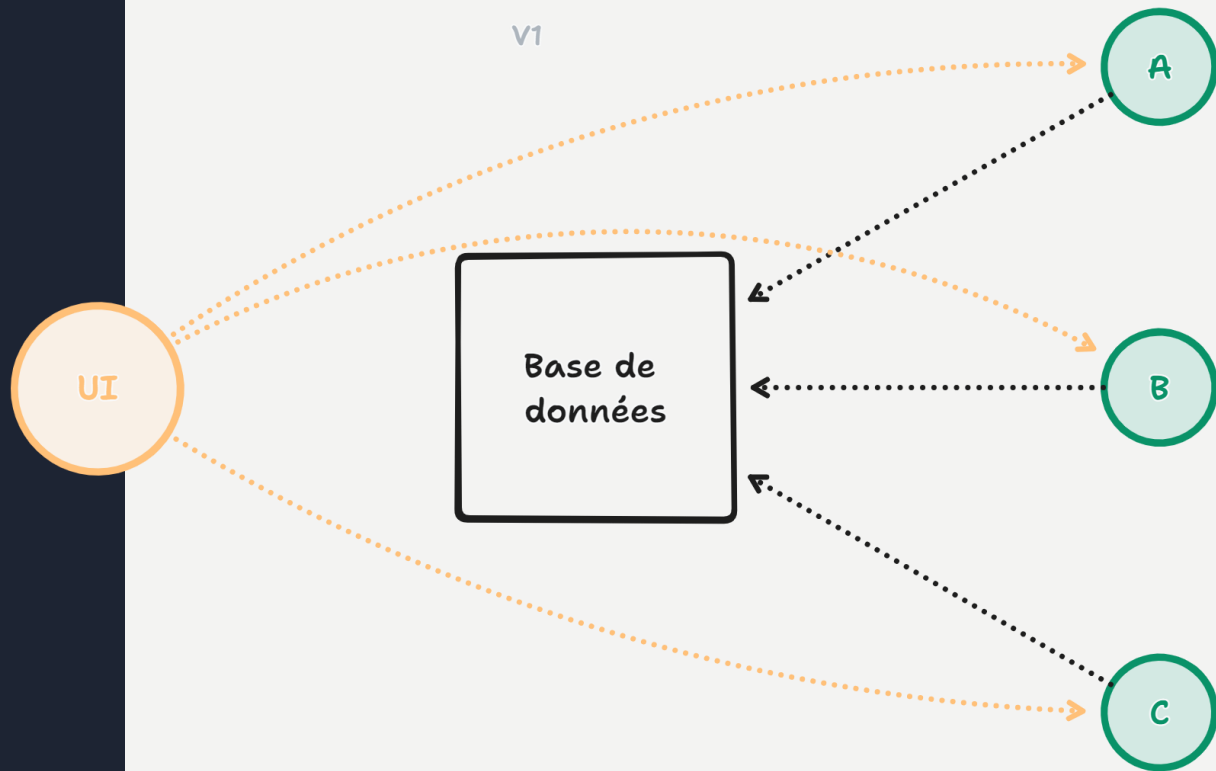
- Utilisation du Pattern **Composition API**
 - L'idée, c'est de récupérer le résultat de sous-requêtes auprès des services concernés
 - Pour ensuite composer une réponse complète, avec toutes les informations nécessaires
 - Parce que quelque chose d'aussi simple ne sera plus possible avec une base de données par service

```
SELECT pro_id, pro_nom, com_note  
FROM produit  
JOIN commentaire ON com_produit_id = pro_id  
WHERE ...
```

BDD PAR SERVICE



BDD PAR SERVICE



BDD PAR SERVICE

- Mettre en place ce changement
 - Le service **Produit** appellera le service **Commentaire** pour avoir la note
 - Le service **Commentaire** appellera le service **Produit** pour avoir le nom

A decorative wavy line in light blue and white, running vertically along the left side of the slide.

SPRING BOOT

EXEMPLE D'IMPLÉMENTATIONS

SPRING BOOT

- Utiliser RestTemplate et l'injection de dépendance

```
@Bean  
public RestTemplate restTemplate() {  
    return new RestTemplate();  
}
```

```
@Autowired  
private RestTemplate restTemplate;
```

```
restTemplate  
    .getForObject("http://host:port/api/...", String[].class);
```

SPRING BOOT

NOTE Penser à consulter start.spring.io
Pour récupérer les dépendances nécessaires

- Utiliser **OpenFeign**
 - Avec la dépendance *spring-cloud-starter-openfeign* de **SPRING CLOUD**
 - L'annotation `@EnableFeignClients`
 - La création d'une interface dédiée

```
@FeignClient(value = "nom-service", url = "http://host:port/")
public interface ServiceClient {
    @GetMapping("/api/endpoint/by-produit-id/{produitId}")
    public List<String> uneRequete(@PathVariable String produitId);
}
```

- L'injection de la dépendance `ServiceClient`

```
serviceClient.uneRequete("id-du-produit");
```