

Cours de bases de données,
<http://sql.bdpedia.fr>

Transactions sur la base "Messagerie"

Transactions sur la base Messagerie

```
# Tous les messages non envoyés
messages = connexion.cursor()
messages.execute("select * from Message where dateEnvoi is null")
for message in messages.fetchall():
    # Marquage du message
    connexion.begin()
    maj = connexion.cursor()
    maj.execute ("Update Message set dateEnvoi='2018-12-31' "
                + "where idMessage=%s", message['idMessage'])

    # Ici on envoie les messages à tous les destinataires

    # Ici on valide
    connexion.commit()
```

Est-elle à l'abri de tout problème de concurrence ?

Analyse de la transaction

On crée une transaction par message. Chaque transaction :

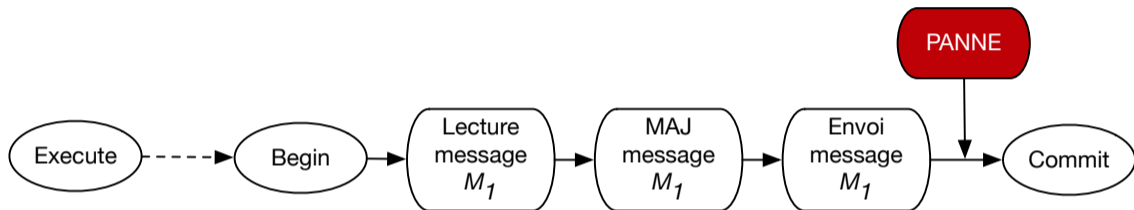
- Lit un message sans date d'envoi dans le curseur
- Modifie le message dans la base en affectant la date d'envoi
- Envoie le message
- Valide (`commit`)

Si chaque transaction s'exécutait seule et sans erreur, chaque message serait envoyé exactement une fois.

Regardons de plus près...

Que se passe-t-il en cas de panne ?

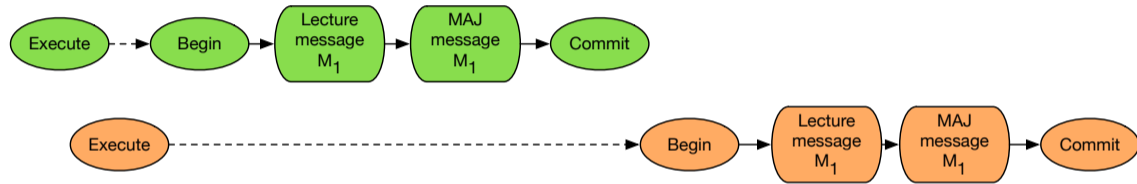
Imaginons (pire scénario) une panne **juste avant** le **commit**



- Le système effectue un **rollback** : le message se retrouve sans date d'envoi (à **null**)
- Le message a quand même été envoyé (le SGBD n'a aucun contrôle là-dessus).

Isolation : et si une autre transaction modifie le message

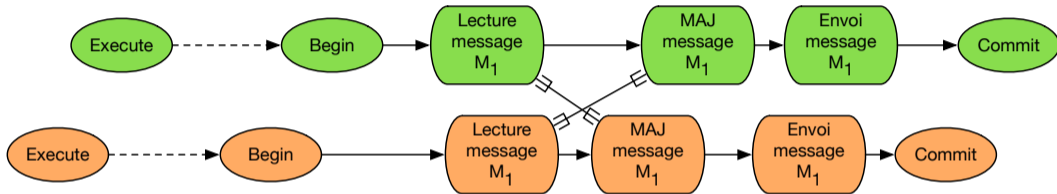
Deux exécutions concurrentes du programme d'envoi



- Le résultat du curseur est constitué au moment du execute
- On continuera à lire des messages même s'ils ont été modifiés ou détruits entretemps !

Isolation et exécutions simultanées

Deux scripts d'envoi de message sont envoyés **presque** en même temps



- **Isolation totale** (mode **serializable**) : on aboutira à des *deadlocks* (les lectures bloquent les écritures)
- Sinon (autres modes), pas de blocage, et défaut d'isolation : les messages seront envoyés deux fois...

À retenir

Toujours utile de se poser la question “Comment ma transaction se comporte-t-elle en cas de concurrence”

- En cas de doute il est préférable de se mettre en mode `serializable`
- Les requêtes fournissant des données à une transaction doivent être **dans** la transaction.

On pourrait envoyer de l'argent au lieu de simples messages, et la question deviendrait très sensible...