SGBD Actifs et Déclencheurs (Triggers)

Khalid Nafil

k.nafil@um5s.net.ma

Année universitaire : 2008/09

Introduction

- Un SGBD actif est capable de réagir à des événements afin de :
  - Contrôler l'intégrité
  - Gérer des redondances
  - Autoriser ou interdire des accès
  - Alerter des utilisateurs

Introduction…

- Une BD active répercute les effets de m-à-j sur certaines tables vers d'autres tables
- Pour réagir, le SGBD pourra :
  - Déclencher une opération subséquente à un événement
  - Interdire une opération
  - Envoyer un message
  - …

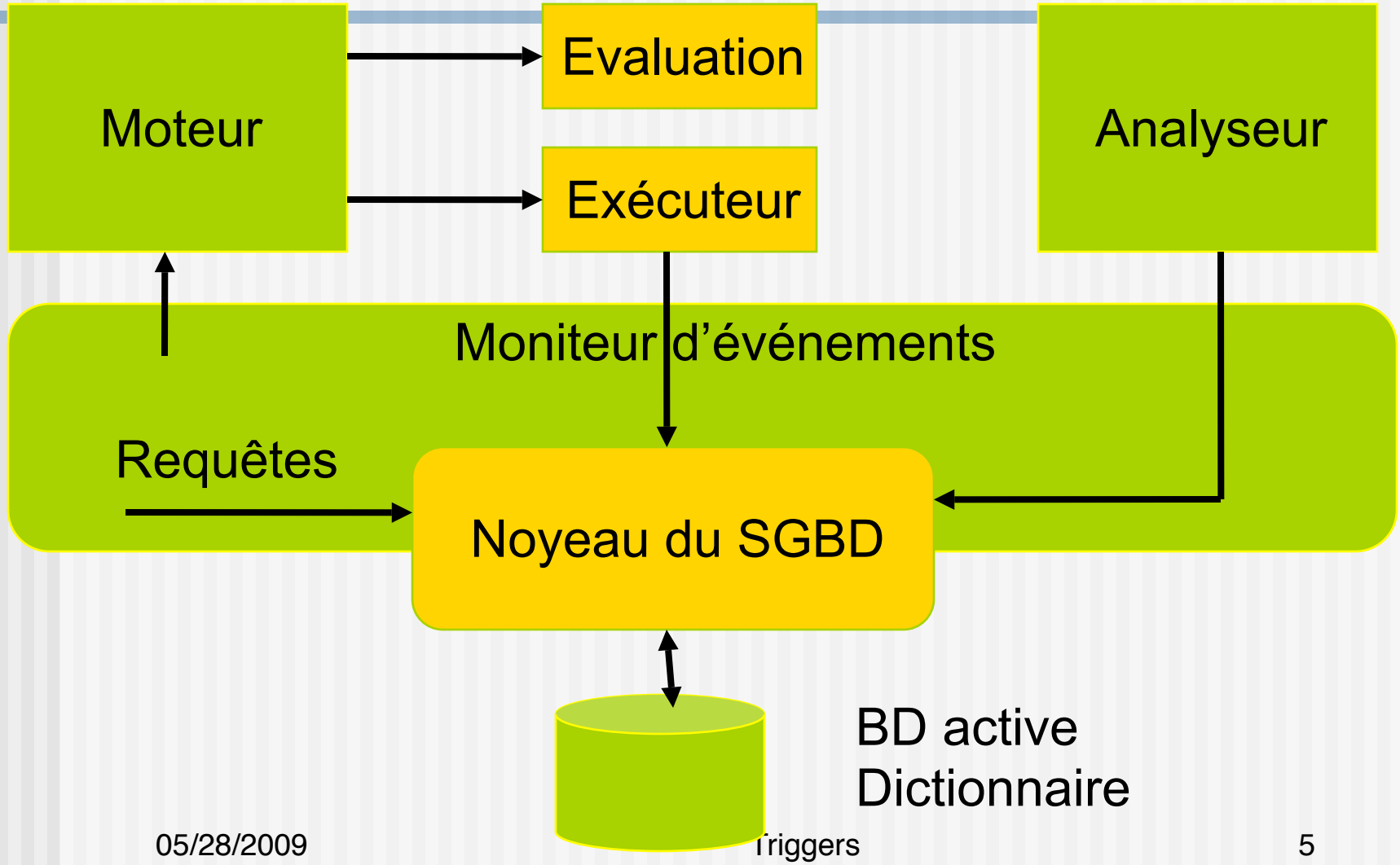
Types de règles

- Les SGBD actifs ont intégré les règles de production de l'IA
- Une règle de production est une construction de la forme :

  If <condition sur BD>

  Then <action sur BD>
- Chaque règle sera appliquée suite à un événement
- Les règles deviennent alors des déclencheurs

Architecture d'un SGBD actif



Composants d'un SGBD actif

- Analyseur de règles
- Moteur de règles
- Moniteur d'événements
- Évaluateur de conditions
- Exécuteur d'actions
- Dictionnaire

Analyseur de règles

- Saisir les règles en mode externe
- Analyser les règles
- Ranger les règles en format interne dans le dictionnaire

Moteur de règles

- Coordonne l'exécution des règles suite aux événements
- Détermine les règles candidates à l'exécution
- Parmi les règles actives, il choisit la plus prioritaire
- Lance son exécution

Moniteur d'événements

- Détecte les événements primitifs et composites
- Demande au moteur de règles l'exécution des règles déclenchées par ces événements

Évaluateur de conditions

- Évalue les conditions des règles actives, sur demande du moteur de règles
- Détermine si la condition est vraie
- Retourne cette information au moteur de règles

Exécuteur d'actions

- Exécute les actions des règles actives dont les conditions ont été préalablement vérifiées

Dictionnaire

- Contient la définition des règles en format interne

Les événements

- Un événement est un signal instantané apparaissant à un point spécifique dans le temps, externe ou interne, détecté par le SGBD
- La spécification d'un événement nécessite la définition d'un type

Types d'événements

- Un type d'événement peut correspondre au :
  - début ou à la fin d'une :
    - Recherche (select)
    - Mise à jour (update, delete, insert)
    - Transaction (begin, commit, abort)
  - Écoulement d'un délai
  - Passage d'une horodate
  - …
- Parmi les événements, on distingue ceux qui sont simples et ceux qui sont composés

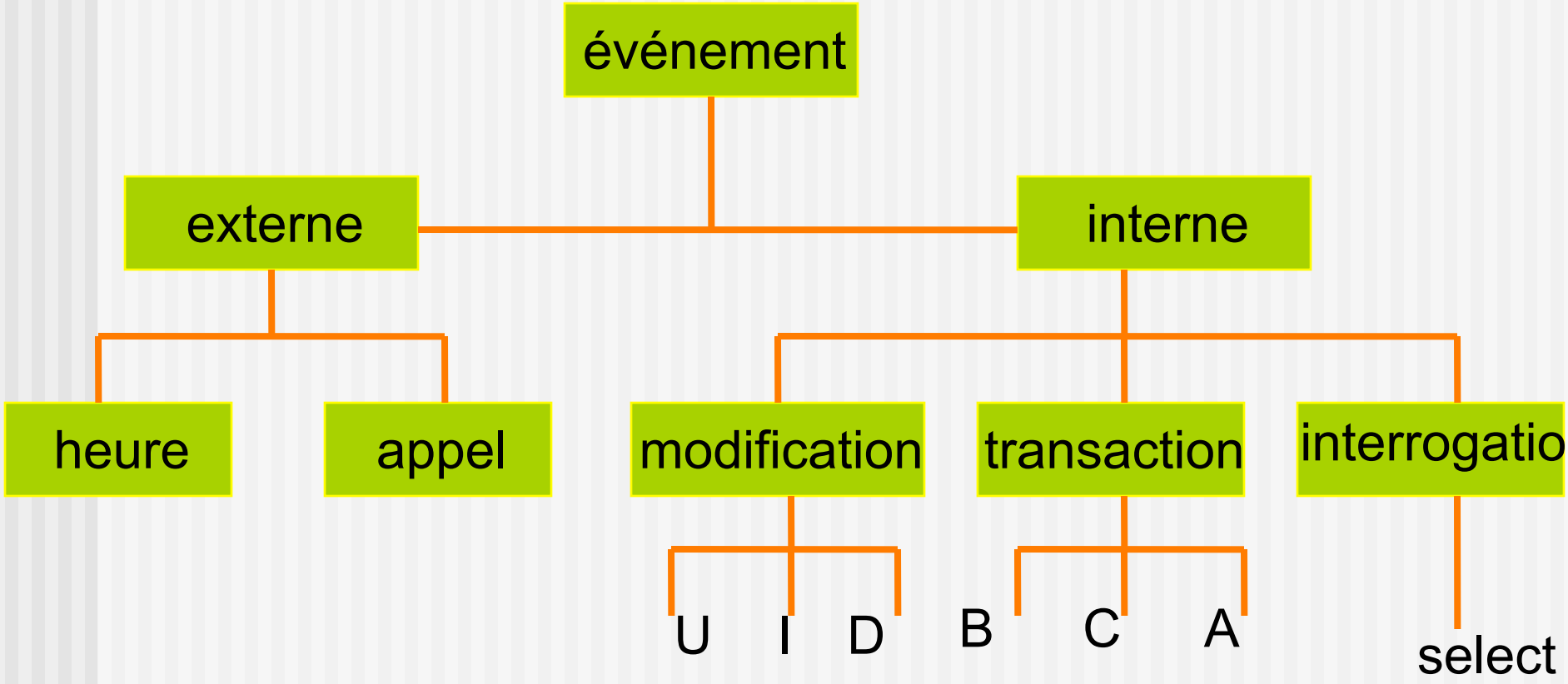
Événements simples (primitifs)

- Appel d'une modification de données (before update ou insert ou delete)
- Terminaison d'une modification de données (after update ou insert ou delete)
- Appel d'une recherche de données (before select)
- Fin d'une recherche de données (after select)

Événements simples…

- Début, validation, annulation d'une transaction (begin, commit, abort)
- Un événement temporel absolu (at times <heure>)
- Un événement temporel relatif (in times <delta>
- Un événement utilisateur (avant ou après exécution de procédure)

Typologie des événements primitifs



Événement composé

- Composition d'événements simples
- La composition est effectuée par des constructeurs spécifiques tel que le Ou et le Et logiques
- Possibilité de faire intervenit des constructeurs temporels tels que A puis B, N fois A, A in <intervalle>
- Peu de SGBD utilisant des événements composés

Condition et Action

- La condition est une expression logique portant sur les variables de contexte d'exécution de la règle ou/et sur la BD
- L'action est une procédure exécutée lorsque la condition est vérifiée

Syntaxe de création de déclencheurs

Create Trigger <nom>

{before | after | instead of}

{insert | delete | update[of <liste colonne>]}

On <table> [order <valeur>]

[referencing {new | old | new_table | old_table} as <nom>]…

(when (<condition de recherche SQL>)

<procedure SQL3>

For each {row | statement}])

Sémantique

- L'événement de déclenchement est une m-à-j, une insertion ou une suppression dans une table
- L'instant d'activation de la règle est défini par 3 options : before, after, instead of
- Referencing new as <nom> définit une variable nommée <nom> contenant la valeur de la dernière ligne modifiée ds la BD par l'événement
- Referencing old as <nom> définit une variable nommée <nom> contenant la valeur de la même ligne avant l'événement

Exemple de Trigger (contrôle d'intégrité)

- Employe(id int, nom varchar, salaire float)
- CIR : le salaire ne peut que croître
- Create trigger salairecroissant

Before update of salaire on employe

Referencing old as O, new as N

(when O.salaire > N.salaire

Signal.sqlstate '7005' ('les salaires ne peuvent décroître)

For each row);

Exemple Trigger (m-à-j automatique de colonnes)…

- Produits(np int, nf int, cout real, auteur string, datemaj date)
- Create trigger SetAuteurDate

Before update on produits

Referencing new_table as N

(update N set N.auteur=user, N.datemaj = currentdate);

- Positionne les attributs auteur et datemaj aux valeurs courantes de la transaction pour lequel il est exécuté

Exemple Trigger…

- Create trigger SetCle

Before insert on produits

Referencing new_table as N

(update N set n.np = select count(*)+1 from produits);

- Crée automatiquement la clé lors de l'insertion d'un tuple

Exemple de trigger (gestion de données agrégatives)

- Cumul (id int, augmentation float)
- Create trigger cumulsalaire

  after update of salaire on employe

  referencing old as a, new as n

  (update cumul set augmentation = augmentation + n.salaire-a.salaire where id=a.id

  for each row);

Procédures déclencheurs sous PostgreSql

- Pl/pgSql peut être utilisé pour définir ces procédures
- Une procédure déclencheur est crée avec la commande create function, sans arguments, et le type retourné est trigger
- Quand une fct Pl/pgsql est appelée comme trigger, plsrs variables spéciales sont crées automatiquement

Variables spéciales

- New : new database for insert/update
- Old : old database for update/delete
- Tg_name : contient le nom du trigger
- Tg_when : une chaine before ou after
- Tg_level : une chaine pour soit row ou statement
- Tg_op : une chaine insert, update ou delete

Variables spéciales…suite

- Tg_relid : l'Id objet de la table causant l'invocation du trigger
- Tg_table_name : nom de la table causant l'invocation du trigger
- Tg_nargs : nombre d'arguments passés dans la procédure déclencheur
- Tg_argv[] : les arguments à partir du create trigger statement
- Une fct déclencheur doit retourner soit null ou un record/row ayant la même structure que la table du déclencheur

Exemple de procédure trigger

- Create table employe(mate text, salaire integer, ancienne_date timestamp, dernier_utilisateur text);
- Create function empl() returns trigger as $empl$

  begin

  if new.mate is null then

  raise exception ‘le matricule ne peut etre null’;

  end if;

Exemple…suite

```
if new.salaire is null then raise exception ‘on ne peut avoir un salaire null (%)’,new.mate;
end if;
new.ancienne_date:=current_timestamp;
new.dernier_utilisat:=current_user;
return new;
end;
$empl$ language plpgsql;
```

- Create trigger empl before insert or update on employe for each row execute procedure empl();