Databázové systémy

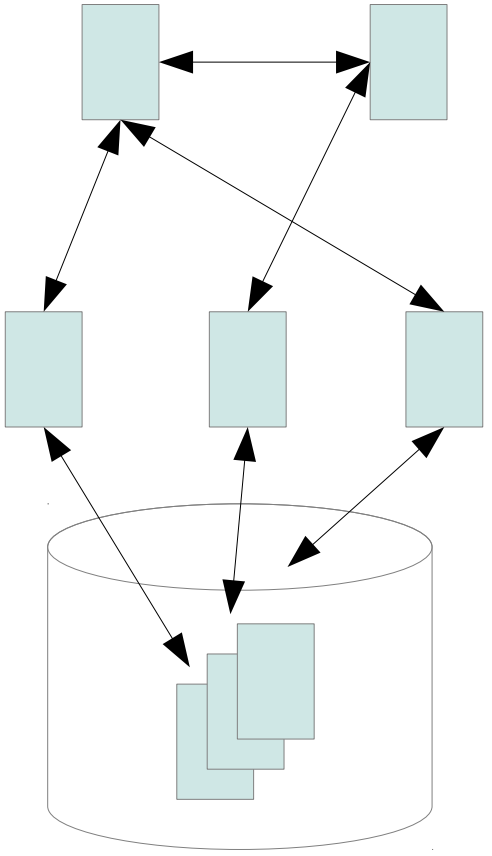
Pohľady - Views

Tri úrovne pohľadu na databázu

logická

konceptuálna

fyzická



Motivácia pre pohľady

- Riadiť prístup k jednotlivým častiam dát
  - ukryť niektoré dáta pred niektorými používateľmi
- zjednodušiť niektoré dopyty
- mať prístup k databáze viac modulárny, flexibilný
  - predstavte si to ako interface – ak sa zmení štruktúra dát pod pohľadom, s aplikáciou to nezamáva

Definícia View

- V = ViewQuery(R_1, R_2,..., R_N)
- Dopyt Q, ktorý pracuje s pohľadom V
  - konceptuálne: temp. tabuľka V, Q pracuje s V
  - reálne je Q prepísané
    - aby používalo R_1, R_2,..., R_N
    - robí to automaticky RDBMS
  - R je relácia ==> môže byť aj iné view
- View nie je uložené, vykoná sa v čase dopytu

Syntax

```
CREATE VIEW view_name
      [(column_list)] AS <Query>
```

demo

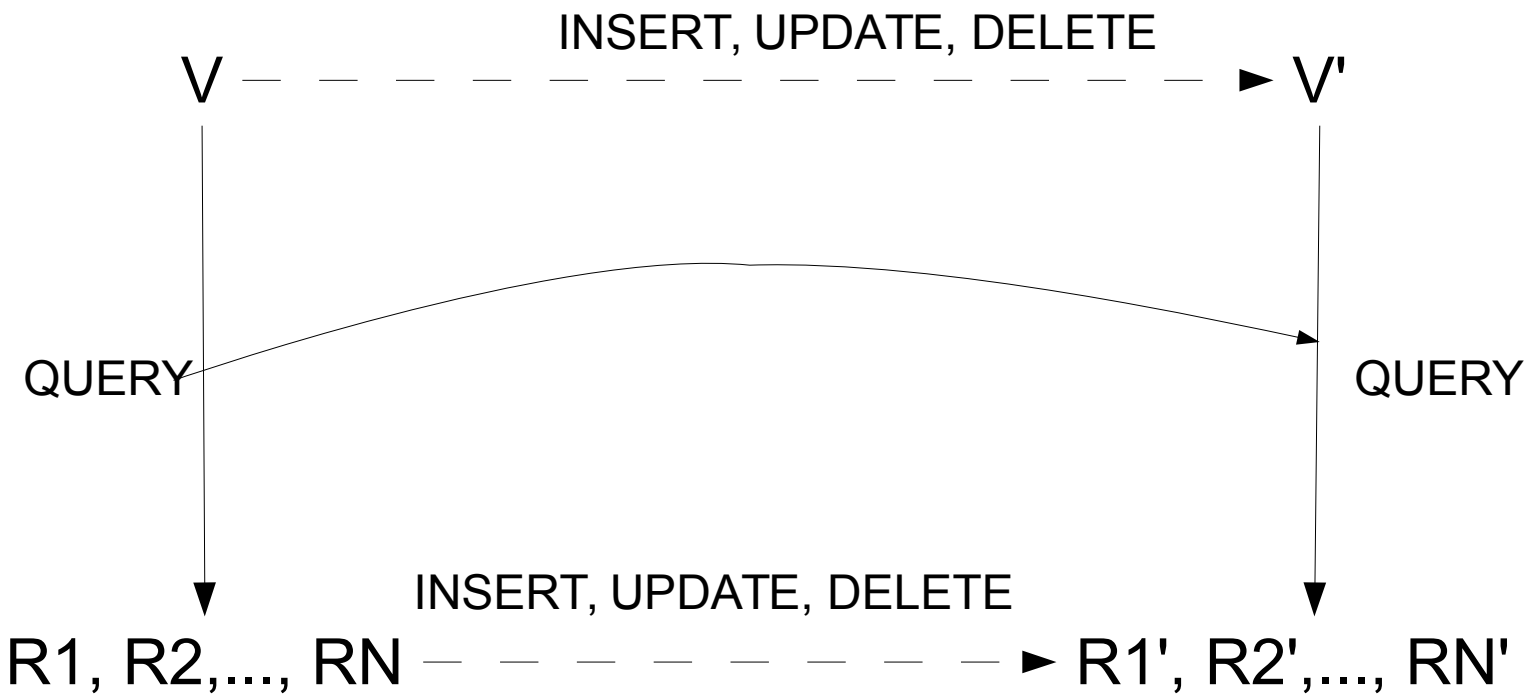
ukážka dopytovania sa do view

Modifikácia údajov vo view

- Môžeme meniť dáta vo view?

- Nie, view predsa nie je nikde uložené
- Áno, view je to jediné, čo možno vidí používateľ
  - ani o tom nevie, že pracuje s view
  - … ale dozvie sa :)

Modifikácia údajov view



Problém

- Viacero možností ako vykonať zmenu v základných tabuľkách
  - R(A,B), V = View(A) FROM R,
    INSERT INTO V VALUES(2);
  - R(N), V = View(avg(R))
    UPDATE V SET avg = 7;

Riešenia

- Spôsob prepisovania základných tabuliek určí tvorca view
  - vieme zvládnuť ľubovoľné modifikácie dát
  - nemáme garancie korektnosti
    - tvorca view je len človek :)
- Obmedziť možnosti modifikácií tak, aby bol preklad jednoznačný a zmysluplný
  - nepotrebujeme tvorcu view
  - máme však zásadné obmedzenia

Riešenie cez triggre

- Napíšte si vlastný INSTEAD OF trigger
  - ale nie v MySQL :)

```
CREATE TRIGGER dobreRestauracieDelete
INSTEAD OF delete ON dobreRestauracie
FOR EACH ROW
BEGIN
  DELETE FROM restaurants r where r.id = Old.id
END;
```

Kedy je view automaticky updatovateľné?

- SELECT (bez DISTINCT) nad jedinou tabuľkou T
- atribúty, ktoré nie sú zahrnuté vo view môžu byť NULL alebo majú default hodnotu
- subquery nesmie referencovať tabuľku T
- bez group by alebo agregácie

- Toto na oplátku podporuje iba MySQL
  - PostgreSQL a SQLite majú INSTEAD OF trigger

demo

- pokusy s MySQL

Materializované pohľady

- cieľ je zvýšiť performance

- oproti klasickému view sa pri tomto type výsledok “materializuje” a uloží sa
- treba ho teda vypočítať znova, ak sa zmenia dáta v tabuľkách, s ktorými view pracuje
  - Nepripomína vám to indexy? Malo by...

- a MySQL to nemá :)

syntax

```
CREATE MATERIALIZED VIEW table_name
    [ (column_name [, ...] ) ]
    [ WITH ( storage_parameter [=
value] [, ... ] ) ]
    [ TABLESPACE tablespace_name ]
    AS query
    [ WITH [ NO ] DATA ]
```

Zhrnutie

- Views poskytujú alternatívne pohľady na tabuľky
- Každú aplikáciu, používateľa môže zaujímať niečo iné
- Modularizujú db, resp. prácu s ňou
- Sprehľadňujú dopyty

- Pri modifikovaní dát to môže byť komplikované