Сигурност података и ауторизација

Ненад Митић

Математички факултет
nenad.mitic@matf.bg.ac.rs

Увод

Сигурност и интегритет

- Сигурност: заштита података од неауторизованих корисника (заштита против неауторизованог приступа, промене или уништења)
- Интегритет: заштита података против ауторизованих корисника (обезбеђење исправности и коректности података) - већ разматрано

Увод - наставак

Сличности између сигурности и интегритета

- систем мора да буде свестан извесних ограничења које корисници не смеју да прекрше
- ограничења морају да буду задата (од стране ДБА) у неком језику
- ограничења морају да буду евидентирана у системском каталогу (речнику података)
- СУБП мора да врши надзор над операцијама корисника

Сигурност

Аспекти проблема сигурности

- правни, социјални и етички (нпр. увид у стање рачуна корисника)
- физичка контрола (нпр. физичко обезбеђење рачунарске сале)
- политичка питања (одлучивање ко и чему сме да приступи)
- оперативни проблеми (нпр. како обезбедити тајност лозинки)

Сигурност (наставак)

Аспекти проблема сигурности (наставак)

- хардверска контрола (нпр. да ли процесор има могућност хардверске заштите програма)
- подршка оперативног система (нпр. да ли ОС брише садржај меморије и дискова по завршетку рада програма)
- проблеми везани за саме базе података (нпр. да ли постоји концепт власника података)

Сигурност (наставак)

Јединица података на коју се осигурава

- база података
- релвар
- појединачна торка или вредност атрибута
- алиаси
- шеме
- индекси
- пакети
- простори за чување табела
- .....

Сигурност у SQL-у

Сигурност се обично записује преко контролне матрице приступа

- представљање по корисницима
- представљање по објектима
- представљање по дозволи за приступ

Пример: видети ауторизације у DB2 у програму Data Studio

Сигурност u SQL-u

Постоје два механизма који су, независно један од другог, укључени у систем заштите

- *Погледи* који могу да се користе за сакривање осетљивих података од неауторизованих корисника
- *Подсистем за ауторизацију*, који допушта кориснику са одређеним правима приступа да та права селективно и динамички преноси на друге кориснике, и/или да пренета права повуче

Погледи - увод

- Релациони системи подржавају **погледе** - именоване релвар-е
- Вредност погледа је резултат извршавања одређеног релационог израза у том тренутку
- Релациони израз се наводи при формирању погледа

Погледи - увод

- Систем претвара упит наведен при формирању погледа у еквивалентан упит над основним релвар-има
- Претварање се врши супституцијом
- Супституција је могућа на основу особине релационог затворења

Примери

```
VAR UPISANI2010 VIEW
(DOSIJE WHERE INDEKS/10000=2010)
 {ALL BUT MESTO_RODJENJA}
RENAME GOD_RODJENJA AS GODINA
```

Поглед upisani2010 има атрибуте Indeks, Ime, Prezime i Godina

Primeri

Поглед који садржи парове назива рока и предмета у којима је предмет полаган

```
VAR PAR_NAZIVA VIEW
((PREDMET RENAME NAZIV AS PNAZIV)
 JOIN ISPIT JOIN
 (ISPITNI_ROK RENAME NAZIV AS INAZIV)
) {PNAZIV, INAZIV}
```

Примери

Поглед може да буде дефинисан и над другим погледом а не само над основним релаваром. Пример: дефинисати поглед који садржи студенте уписане 2010 године који су рођени после 1992. године

```
VAR UPISANI_2010_1992  VIEW
UPISANI2010 WHERE GODINA>1992
```

Формирање погледа

Синтакса наредбе за формирање погледа

```
VAR <име релвар-а> VIEW <релациони израз>
 <листа кандидата за кључеве>
[RESTRICT/CASCADE]
```

- <листа кандидата за кључеве> може бити и празна ако поглед може да наследи кандидате за кључеве
- RESTRICT/CASCADE може али не мора да постоји

Брисање погледа

Синтакса наредбе за брисање погледа

```
DROP VAR <име релвар-а>
```

Особине погледа

Дефиниција погледа комбинује

- спољашњу шему
- пресликавање између спољашњег и концептуалног нивоа (садржи изглед спољашњег објекта и опис како се он пресликава на концептуални ниво)
- спољашње/спољашње пресликавање (нпр. поглед `UPISANI_2010_1992`)

Функције погледа

- Обезбеђују аутоматску заштиту за скривене податке
- Омогућују да различити корисници (истовремено) виде исте податке на различите начине
- Упрошћавају сложене операције (слично макроима у програмским језицима)
- Омогућују логичку независност података

Логичка независност података

- Проширење релација базе не сме да ефекат на извршавање апликативних програма
- Реструктуирање базе не сме да има ефекат на постојеће апликативне програме. Нова и стара база треба да буду информационо еквивалентне.

Логичка независност података

Пример реструктурирања базе: ако се релација DOSIJE разбије на две основне релације

```
VAR DOSIJE1 BASE RELATION
    {INDEKS INDEKS, IME IME,
     PREZIME PREZIME, GOD_RODJENJA INTEGER}
PRIMARY KEY {INDEKS}

VAR DOSIJE2 BASE RELATION
    {INDEKS INDEKS, MESTO_RODJENJA CHAR}
PRIMARY KEY {INDEKS}
```

Логичка независност података

`DOSIJE` може да се формира као поглед

```
VAR DOSIJE VIEW  DOSIJE1 JOIN DOSIJE2
```

Апликативни програми који су радили са основним релвар-ом `DOSIJE` би требало да могу без измена да се реферишу на поглед `DOSIJE`

Читање преко погледа

- Операција читања података преко погледа се конвертује у еквивалентну операцију над основним релвар-има
  - материјализација релације која је тренутна вредност погледа
  - супституција релационог израза у другом релационом изразу
- Семантика погледа се дефинише преко материјализације релација!

Ограничења интегритета код погледа

- Златно правило се примењује и на погледе, тј. ажурирање погледа не сме да наруши ограничења интегритета над погледима
- Ограничења интегритета над погледима су изведена из ограничења интегритета основних релвар-а

Ажурирање погледа

При ажурирању неког погледа какве врсте ажурирања треба извести над основним релацијама да би се имплементирало оригинално ажурирање погледа?

Ажурирање погледа

Прецизније, ако је D база, V поглед над D и X функција над D којом се дефинише поглед V, тада за дати поглед $V = X(D)$ и операцију ажурирања U над V, потребно је одредити операцију ажурирања U_1 над D тако да важи $U(X(D)) = X(U_1(D))$

Могуће је да постоји више операција U_1. Коју одабрати?

Ажурирање погледа

- *Codd*-ов приступ: дефинисање погледа који могу да се ажурирају
- *Date*-ин приступ: сви погледи могу да се ажурирају. Операције ажурирања се изводе избегавањем ограничења интегритета у међукорацима ажурирања
  - ажурирање се изводи као брисање постојећих и уношење нових података

Формирање и брисање

CREATE VIEW наредба/ DROP VIEW наредба/

```
CREATE [OR REPLACE ] VIEW
      <име погледа> AS <израз над табелом>
[WITH [<квалификатор>] CHECK OPTION]
```

```
DROP VIEW <име погледа>
```

Примери дефиниције погледа

Формирати поглед prosek_ocena који садржи индекс, име и презиме студента и тренутну вредност његове просечне оцене

```
create or replace view prosek_ocena
          (indeks, ime, prezime, trenutni_prosek)
as
select a.indeks, b.ime, b.prezime, dec(avg(ocena*1.0),4,2)
from   ispit a, dosije b
where  a.indeks=b.indeks
and    a.ocena>5
group  by a.indeks, b.ime, b.prezime;
```

Примери дефиниције погледа

Формирати поглед који садржи, за свако место рођења из табеле досијеа, податке само за једног студента који је рођен у том месту

```
create or replace view predstavnik as
with
isti(indeks1, indeks2) as
(
  select a.indeks, b.indeks
  from   dosije a, dosije b
  where a.mesto_rodjenja=b.mesto_rodjenja and a.indeks<b.indeks
),
jedinstveni(indeks) as
(
 select distinct a.indeks1
 from isti a
 where   indeks1 not in (select indeks2 from isti b)
)

select  indeks,ime,prezime, mesto_rodjenja as mesto, god_rodjenja as godina
from    dosije
where   indeks in (select indeks from jedinstveni)

union all

select  indeks,ime,prezime, mesto_rodjenja as mesto, god_rodjenja as godina
from    dosije
where   indeks not in (select indeks1 from isti)
and     indeks not in (select indeks2 from isti);
```

Издвајање резултата

- По стандарду би све операције са читањем података преко погледа требало да раде коректно
- У пракси то није случај, поготову ако се погледи не материјализују

Издвајање резултата

Пример: упит

```
select avg(ispit.zbir) from ispit
```

уколико се не материјализује се имплементира као

```
select avg(sum(ispit.ocena))
from   ispit
group  by id_predmeta
```

који је некоректан јер садржи угнеждену агрегатну функцију.

Ажурирање погледа

- SQL/92 подршка за ажурирање погледа је врло ограничена
- Једини погледи који се сматрају могућим за ажурирање су погледи који су изведени из једне основне табеле преко комбинација рестрикције и пројекције

Ажурирање погледа

У SQL/92 поглед може да се ажурира ако важи

- Израз којим се дефинише поглед је SELECT израз који не садржи JOIN, UNION, INTERSECT или EXCEPT
- Select клаузула не садржи кључну реч DISTINCT
- Свака SELECT ставка садржи (квалификовано) име које представља референцу на колону основне табеле

Ажурирање погледа

- *from* клаузула садржи референцу на тачно једну табелу која је или основна табела или поглед који може да се ажурира
- *where* клаузула select израза не садржи подупит у коме се from клаузула реферише на исту табелу као и from клаузула у select изразу на највишем нивоу
- *select* израз не садржи group by нити having клаузулу

Погледи у DB2 V11.5

INSTEAD OF опција код дефиниције окидача (триггера) омогучује индиректно ажурирање погледа који према дефиницији могу само да се читају.

За дискусију о погледима који могу да се ажурирају, у које могу да се уносе подаци и из којих могу да се бришу подаци у DB2 видети

1. *Db2 V11.5 SQL Reference* str. 1520
2. https://www.ibm.com/docs/en/db2/11.5?topic=triggers-instead

Sigurnost u SQL-u - Подсистем за ауторизацију

- Да би се корисник извршио било какаву операцију над неким објектом он мора да поседује дозволу (или ауторизацију) за ту операцију над тим објектом
- Типови и врсте дозвола нису исти код свих СУБП
- Системски администратор (SYSADM ниво ауторизације) је иницијални власник свих дозвола
- Давање дозвола се врши GRANT наредбом, а повлачење REVOKE наредбом
- Комплетна синтакса у DB2 SQL Reference

GRANT наредба

Синтакса за дозволе над табелама или погледима

GRANT дозвола [ON [тип] објекат] TO корисник;

дозвола је листа једне или више врста дозвола, раздвојеним зарезима или фраза ALL PRIVILEGES или ALL (која означава све привилегије које се могу дати GRANT наредбом

корисник је листа корисника раздвојених зарезима или PUBLIC (сви корисници)

објекат је листа имена једног или више објеката (који су сви истог типа) раздвојених зарезима

тип означава тип објекта - ако се изостави подразумева се TABLE

ON се не употребљава када се даје дозвола са системске привилегије

GRANT наредба

Дозволе које се односе на основне табеле и погледе

GRANT дозвола [ON [тип] објекат] TO корисник;

- CONTROL
- DELETE
- INSERT
- SELECT
- UPDATE (могу да се наведу појединачне колоне)

GRANT наредба (наставак)

Дозволе које се односе само на основне табеле

- ALTER (дозвола за извршавање ALTER TABLE над табелом)
- INDEX (дозвола за извршавање CREATE INDEX над табелом)
- REFERENCES (дозвола за формирање/брисање спољашњег кључа који реферише ту табелу као родитељ табелу)

GRANT наредба - примери

Примери дозвола за операције над табелама

- GRANT SELECT ON TABLE DOSIJE TO KORISNIK01;
- GRANT SELECT, UPDATE (SIFRA, NAZIV) ON TABLE PREDMET TO KORISNIK02, KORISNIK03, KORISNIK09;
- GRANT ALL PRIVILEGES ON TABLE DOSIJE, PREDMET, ISPIT TO KORISNIK76, KORISNIK77;
- GRANT SELECT ON TABLE DOSIJE TO PUBLIC;
- GRANT DELETE ON ISPITNI_ROK TO KORISNIK99;

REVOKE наредба

Синтакса за дозволе над табелама или погледима

REVOKE дозвола [ON [тип] објекат] FROM корисник [BY ALL];

REVOKE наредба (наставак)

Повлачење дозволе за неког корисника узрокује да сви планови/пакети засновани на тој дозволи постану неисправни и узрокују аутоматско везивање/поновно везивање приликом позивања таквог плана/пакета.

Није могуће укинути UPDATE дозволу само за поједине колоне.

GRANT наредба - WITH GRANT OPTION

Ако корисник K1 жели да пренесе дозволу D кориснику K2, то може да уради

- наредбом GRANT дозовла ....
- наредбом
  GRANT дозвола ... WITH GRANT OPTION
  чиме омогућује кориснику K2 да даље прослеђује дозволу која му је пренета

WITH GRANT OPTION - примери

Neka je dato

Корисник K1:
GRANT SELECT ON TABLE DOSIJE TO K2 WITH GRANT OPTION;

Корисник K2:
GRANT SELECT ON TABLE DOSIJE TO K3 WITH GRANT OPTION;

Корисник K3:
GRANT SELECT ON TABLE DOSIJE TO K4 WITH GRANT OPTION;

.....

WITH GRANT OPTION - примери

Тада повлачење дозволе

REVOKE SELECT ON TABLE KVOTA FROM K2;

проузрокује ланчано повлачење дозволе за кориснике K3, K4, ...

Д.З. Проверити да ли у претходном случају ако је корисник K3 (K4) имао и дозволу добијену од неког другог корисника и та дозвола повучена или је остала важећа

Остали аспекти сигурности

Комплетан систем треба да буде заштићен

Не претпостављати да је систем заштите савршен

Водити евиденцију о пријављивању на базу

Ауторизација у DB2

Нивои ауторизације (видети SQL Reference)

Примери