Pogledi (views) v SQL

- Pogled (**VIEW**) je tabela, katare vrstice NISO shranjene v podatkovni bazi, ampak se sproti računajo na podlagi *definicije pogleda*.
- Uporaba pogledov: pogosto uporabljane poizvedbe, omejitev dostopa do nekaterih stolpcev, izločevanje nepotrebnih detajlov
- Pogledi so definirani s **SELECT** stavki
- Vsaka sprememba v bazi se pozna v pogledu in obratno: vsaka sprememba v pogledu se pozna v bazi

Kreiranje in brisanje pogledov

- Sintaksa za kreiranje:

```
CREATE VIEW ime_pogleda(imena atributov)
AS SELECT stavek;
```

- Imena atributov lahko izpustimo, v tem primeru so v pogledu vsi atributi rezultata poizvedbe
- Paziti moramo na morebitna podvojena imena atributov in jih po potrebi preimenovati
- Sintaksa za brisanje:

```
DROP VIEW ime_pogleda;
```

Pogledi: rezervacija z barvo

```
CREATE VIEW barv_rez
AS SELECT r.*, c.barva
   FROM coln c, rezervacija r
   WHERE c.cid = r.cid;

SELECT *
FROM barv_rez;
```

jid	cid	dan	barva
22	101	2006-10-10	modra
64	101	2006-09-05	modra
22	102	2006-10-10	rdeca
31	102	2006-11-10	rdeca
64	102	2006-09-08	rdeca
22	103	2006-10-08	zelena
31	103	2006-11-06	zelena
74	103	2006-09-08	zelena
22	104	2006-10-07	rdeca
31	104	2006-11-12	rdeca

Pogledi: coln z rezervacijo (1)

```
CREATE VIEW coln_rez
AS SELECT *
   FROM coln c, rezervacija r
   WHERE c.cid = r.cid;
```

- Problem: dva stolpca z istim imenom (cid)
- Lahko rešimo na tri načine:
  - preimenujemo v SELECT stavku
  - preimenujemo v CREATE VIEW stavku
  - izločimo podvojene atribute

Pogledi: coln z rezervacijo (2)

```
CREATE VIEW coln_rez
AS SELECT r.*, c.cid AS ccid, c.ime, c.barva, c.dolzina
    FROM coln c, rezervacija r
    WHERE c.cid = r.cid;
SELECT * FROM coln_rez;
```

jid	cid	dan	ccid	ime	barva	dolzina
22	101	2006-10-10	101	Elan	modra	34
64	101	2006-09-05	101	Elan	modra	34
22	102	2006-10-10	102	Elan	rdeca	34
31	102	2006-11-10	102	Elan	rdeca	34
64	102	2006-09-08	102	Elan	rdeca	34
22	103	2006-10-08	103	Sun Odyssey	zelena	37
31	103	2006-11-06	103	Sun Odyssey	zelena	37
74	103	2006-09-08	103	Sun Odyssey	zelena	37
22	104	2006-10-07	104	Bavaria	rdeca	50
31	104	2006-11-12	104	Bavaria	rdeca	50

Pogledi: coln z rezervacijo (3)

```
CREATE VIEW coln_rez(jid,cid,dan,ccid,ime,barva,dolzina)
AS SELECT *
    FROM coln c, rezervacija r
    WHERE c.cid = r.cid;
SELECT * FROM coln_rez;
```

jid	cid	dan	ccid	ime	barva	dolzina
22	101	2006-10-10	101	Elan	modra	34
64	101	2006-09-05	101	Elan	modra	34
22	102	2006-10-10	102	Elan	rdeca	34
31	102	2006-11-10	102	Elan	rdeca	34
64	102	2006-09-08	102	Elan	rdeca	34
22	103	2006-10-08	103	Sun Odyssey	zelena	37
31	103	2006-11-06	103	Sun Odyssey	zelena	37
74	103	2006-09-08	103	Sun Odyssey	zelena	37
22	104	2006-10-07	104	Bavaria	rdeca	50
31	104	2006-11-12	104	Bavaria	rdeca	50

Data definition language - DDL

- Tipi atributov
- Kreiranje in spreminjanje tabel
- Polnjenje tabel

Tipi atributov

- Numerični tipi: NUMBER, INTEGER, FLOAT, DOUBLE, DECIMAL, ...
- Znakovni tipi: CHAR, VARCHAR, TEXT, ...
- Datumski tip: DATE
- Netipizirani tipi: BLOB (binary large object), CLOB (character large object) ali TEXT, velikost ene vrednosti v MariaDB do 4GB

Kreiranje tabel

- Sintaksa:

```
CREATE TABLE ime_tabele
    (atributi in omejitve)
    druge opcije;
```

- Primer:

```
CREATE TABLE Jadralec
( jid      INTEGER,          -- atributi
  ime      VARCHAR(10),
  rating   INTEGER,
  starost  REAL,
  PRIMARY KEY (jid),        -- omejitve
  CHECK (rating >= 1 AND rating <= 10 ));
```

Polnjenje tabel

- Sintaksa:
  INSERT INTO ime_tabele **VALUES**(v_1, ..., v_k);
  INSERT INTO ime_tabele(ime_1,...,ime_k) **VALUES**(v_1, ..., v_k);
- V drugem primeru lahko nekatere vrednosti manjkajo in dobijo vrednost **NULL**
- Primer:

```
INSERT INTO Jadralec
       VALUES( 22, 'Darko', 7, 45.0);
INSERT INTO Jadralec(jid, ime, rating, starost)
       VALUES( 29, 'Borut', 1, 33.0);
```

Ustvarjanje in polnjenje tabele hkrati

```
CREATE TABLE MaldoletniJadralci AS
        SELECT  *
        FROM jadralec
        WHERE starost <18;
```

Spreminjanje in brisanje tabel

- Spreminjanje tabel:
  - **ALTER TABLE** ime_tabele opcije;
  - dodajanje, brisanje, preimenovanje in spreminjanje atributov
  - dodajanje, brisanje in spreminjanje omejitev, indeksov, ...
  - Povezava na celotno sintakso je tukaj
- Brisanje tabel:
  - **DROP TABLE** ime_tabele;
  - Tabele je potrebno brisati v pravilnem vrstnem redu glede na omejitve tabele (foreign key)

Praznjene in posodabljanje tabel

- Praznjene:
  DELETE FROM ime_tabele **WHERE** pogoj;
- Posodabljanje:
  UPDATE ime_tabele **SET** atribut=vrednost **WHERE** pogoj**;**
- Primer:

```
DELETE FROM coln WHERE barva IS NULL;
UPDATE coln SET dolzina = 40 WHERE cid = 104;
```

Posebnost: MYSql Workbench uporablja safe mode, ki vam ne dovoli uporabljati ukazov update in delete, če ne podate pogoja, ki izbira vrstice glede na primarni ključ.

Indeksiranje v SQL

- Indeks je za uporabnika nevidna podatkovna struktura, ki bistveno pospeši dostop do vrstic tabele; preiskovanje *n* vrstic: s $t_1=O(n)$ na $t_2=O(\log n)$;
- pri n=1 milijon je $t_1= 1000000$, $t_2= 6$ (stevilo korakov)
- Indeksiramo po enem ali več stolpcih
- Zakaj vedno ne indeksiramo celotne tabele:
  - za *k* atributov je možnih 2^k indeksov (vse podmnožice)
  - vsak indeks zahteva prostor na disku in čas za njegovo gradnjo in posodabljanje ob spremembah tabele

Kdaj zgraditi indeks na podmnožici atributov?

- Ključi in ostali enolični (UNIQUE) atributi: pogosto avtomatsko generiranje
- Pogostost preiskovanja in urejanja
- Velikost tabele
- Porazdelitev podatkov
- Prostorsko-časovne omejitve: prostor na voljo v PB, pogostost spreminjanja tabele

Kreiranje in brisanje indeksov

- Kreiranje indeksov:

```
CREATE [UNIQUE] INDEX ime_indeksa
ON ime_tabele (ime_atributa1 [ASC|DESC],
               ime_atributa2 [ASC|DESC],
               ... );
```

- Indeks se gradi po kombinaciji vrednosti atributov: za vsako kombinacijo atributov potrebujemo svoj indeks
- Možna specifikacija tipa indeksa (npr. BTREE, HASH)
- Brisanje nepotrebnih indeksov:

```
DROP INDEX ime_indeksa ON ime_tabele;
```

Primer indeksiranja

- Indeksiraj čolne po barvi!

```
CREATE INDEX po_barvi
ON coln(barva);
```

- Indeksiraj rezervacije ločeno po datumih ter šifrah jadralcev in čolnov skupaj!

```
CREATE INDEX po_dnevih
ON rezervacija(dan);

CREATE INDEX po_jid_cid
ON rezervacija(jid,cid);
```

Uporaba indeksov

- Indeksi se uporabljajo avtomatsko, ko jih enkrat kreiramo: sistem izbere, katerega od potencialno več možnih obstoječih bo uporabil
- Eksplicitna (ne)uporaba indeksov: dosežemo s specialnimi komentarji ali ukazi - namigi (hints)
- Zakaj namigi? Ker vnaprej vemo več kot sistem o tem, kako se bodo podatki uporabljali
- Namigi so NESTANDARDNA razširitev SQL

Namigi za indeksiranje v MariaDB

- Namig: dodana ključna beseda v SELECT stavku za imenom tabele v FROM vrstici

```
-- Uporabi samo naštete indekse
USE INDEX(ime_indeksa1, ime_indeksa2, ...)

-- Ne uporabi nobenega indeksa
USE INDEX()

-- Ignoriraj naštete indekse
IGNORE INDEX(ime_indeksa1, ime_indeksa2, ...)
```

Primeri nekaterih namigov v MariaDB

- Denimo, da smo vnaprej kreirali indekse `jad_index1(jid,ime), jad_index2(jid), jad_index3(ime).`

```
SELECT *
FROM jadralec
   USE INDEX(jad_index1)      -- uporabi indeks
ORDER BY ime, jid;            -- po jid in imenu

SELECT *
FROM jadralec
   IGNORE INDEX(jad_index1, jad_index2, jad_index3)
ORDER BY ime, jid;  -- ne uporabi nobenega nastetega
                    -- indeksa
```

Vaje

1. Definirajte pogled MladoletniJadralci, ki vsebuje samo jadralce mlajše od 18 let.
2. Definirajte pogled StatistikaColnov, ki bo za vsak čoln izpisal osnovne podatke (šifra, ime, dolžina), število rezervacij, število različnih jadralcev, ki so ga rezervirali in povprečni rating jadralcev, ki so ga rezervirali.
3. Definirajte pogled StatistikaAlians, ki bo za vsako alianso izpisal osnovne podatke alianse (šifra, ime), število igralcev, število naselji in število vseh prebivalcev.

- Če uporabljate bazo na pb.fri.uni-lj.si, si k imenom svojih tabel dodajte svojo vpisno številko.

Vaje

1. Po tabeli rezervacij pogosto preiskujemo po atributih jid in cid posamezno, ter po (jid, cid) skupaj. Kreirajte ustrezne indekse!
2. Kreirajte novo tabelo, ki ne vsebuje mladoletnih jadralcev!
3. Leto je naokoli, postarajte jadralce!
4. Izbrišite jadralce, ki imajo rating pod 5!
5. Tine, ki ima 19 let in ocenjen rating 8 je danes prvič rezerviral čoln „Bavaria“ ter hkrati opravil registracijo jadralca. Dodajte ga v tabelo jadralcev!
6. Jadralsko društvo je ugotovilo, da bi radi jadralcem omogočili rezervacijo čolnov za več dni skupaj. Dopolnite tabelo rezervacij tako, da bo to omogočala. Za jadralce, ki že imajo rezervacijo se privzame, da so rezervacijo opravili samo za en dan.

Vaje*

1. Definirajte pogled StatistikaJadralcev, ki bo za vsakega jadralca poleg njegovih podatkov (šifra in ime) vseboval tudi število rezervacij čolnov, povprečno dolžino in **prevladujočo barvo*** rezerviranih čolnov.
2. Definirajte pogled StatistikaIgralcev, ki za vsakega igralca traviana izpiše njegovo ime, število njegovih naselji, pleme, ki mu pripada, ter središčno točko okoli katere se nahajajo igralčeva naselja ter maksimalni odmik njegovega naselja od te točke.

- Če uporabljate bazo na pb.fri.uni-lj.si, si k imenom svojih tabel dodajte svojo vpisno številko.

Procedure

- Shranjeni podprogrami, ki jih lahko kličemo
- Parametre lahko določimo kot vhodne (IN), izhodne (OUT) ali vhodno-izhodne (IN OUT)
- Kličemo jih s CALL Ime_Procedure(…)

Primer

Napišite shranjeno proceduro, ki vrne vse podatke o čolnih z dolžino med "spodnja" in "zgornja" meja.

```
DELIMITER //
CREATE PROCEDURE colni_razpon (IN spodnja
INTEGER, IN zgornja INTEGER)
BEGIN
        SELECT * FROM coln
        WHERE dolzina BETWEEN spodnja AND zgornja;
END //
DELIMITER ;

CALL colni_razpon(20,40);
```

Dostop do parametrov

- Na primer, da imate proceduro rezervacijeJadralca(IN *ime* VARCHAR(255), OUT *n* INTEGER), ki vam v zadnjem parametru vrne število rezervacij jadralcev z imenom *ime*.

```
CALL rezervacijeJadralca(„Henrik“, @a);
SELECT @a;
```

Funkcije

- Podobno kot procedure, le da vračajo vrednost
- Parametri so privzeto IN
- Uporaba SELECT Ime_Funkcije(…)

Primer

Napišite funkcijo, ki vrne število čolnov z dolžino med "spodnja" in "zgornja" meja.

```
DELIMITER //
CREATE FUNCTION colni_razpon_fun (spodnja
INTEGER, zgornja INTEGER) RETURNS INTEGER
BEGIN
DECLARE X INTEGER;
SELECT COUNT(*) INTO X
FROM coln
WHERE dolzina BETWEEN spodnja AND zgornja;
       RETURN X;
END //
DELIMITER ;
SELECT colni_razpon_fun(20,40);
```

Dostop do rezultata

- Na primer, da imate proceduro rezervacijeJadralca(IN *ime* VARCHAR(255)), ki vam vrne število rezervacij jadralcev z imenom *ime*.

SELECT rezervacijeJadralca(„Henrik“)

Brisanje procedur in funkcij

- DROP PROCEDURE IF EXISTS Ime_procedure;
- DROP FUNCTION IF EXISTS Ime_Funkcije;

Bazni prožilci

- Podobni proceduram, le da nimajo argumentov in se avtomatsko kličejo ob ažuriranju tabele.
- Lahko se kličejo pri vstavljanju (INSERT), brisanju (DELETE) ali posodabljanju (UPDATE).
- Stavčni in vrstični prožilci.

Primer

- Kreirajte tabelo *MladoletniJadralci* s funkcionalnostjo materializiranega pogleda, ki hrani samo jadralce mlajše od 18 let. Dodajte bazne prožilce za vstavljanje, brisanje in posodabljanje vrstic originalne tabele *jadralci*.

Bazni prožilec za vstavljanje

```
DELIMITER //
CREATE TRIGGER MladoletniJadralci_Insert
AFTER INSERT ON jadralec
FOR EACH ROW
BEGIN
IF NEW.starost <18 THEN
  INSERT INTO MladoletniJadralci VALUES(NEW.jid, NEW.ime,
                                        NEW.starost, NEW.rating);
END IF;
END //
```

* Prožilce za brisanje in spreminjanje dodajte sami za vajo.

Vaje

- Napišite funkcijo Prebivalstvo(imeAlianse), ki vam vrne število prebivalcev v aliansi.
- Napišite proceduro Brisi(pop), ki bo izbrisala vse igralce s popupacjo pop ali manj, ter nato še vse alianse, če le ta nima več igralcev.
- Tabeli MladoletniJadralci dodajte še bazne prožilce za brisanje in spreminjanje vrstic iz tabele jadralci.

Vaje

- Napišite proceduro *Zdruzi(aliansa1*, *aliansa2*, *novaAliansa*), ki združi igralce iz alians *aliansa1* in *aliansa2* in ustvari novo alianso *novaAliansa*. Originalni aliansi je potrebno odstraniti.
- Napišite funkcijo povStarost(cid), ki vrne povprečno starost jadralcev, ki so rezervirali čoln s šifro *cid*.