Pogledi (views) v SQL

- Pogled (**VIEW**) je tabela, katere vrstice NISO fizično shranjene 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 pogleda:

```
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
- Uporabimo lahko tudi **`CREATE OR REPLACE VIEW`** za posodabljanje pogleda.
- Sintaksa za brisanje:

```
DROP VIEW ime_pogleda;
```

Pogledi: rezervacija z barvo

```
CREATE VIEW barv_rez
AS SELECT r.*, c.barva
   FROM coln c JOIN rezervacija r USING(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 JOIN rezervacija r USING(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 JOIN rezervacija r USING(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 JOIN rezervacija r USING(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

Indeksiranje v SQL

- Indeks je s strani 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 $t_1= 1000000$, $t_2= 6$ (število korakov)
- Indeksiramo po enem ali več stolpcih skupaj
- 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

Ustvarjanje in brisanje indeksov

- Ustvarjanje 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 ustvarimo; sistem sam 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
```