



Uvod v SQL...

- SQL je transformacijsko usmerjen jezik, ki ga sestavljata dve večji skupini ukazov (in dve manjši):
 - Skupina ukazov DDL (Data Definition Language) za opredelitev strukture podatkovne baze in
 - Skupina ukazov DML (Data Manipulation Language) za poizvedovanje in ažuriranje podatkov (vključuje tudi DQL – Data Query Language).
 - Skupina ukazov DCL – nadzor dostopnih dovoljenj (Data Control Language)
 - Skupina ukazov TCL – nadzor sočasnih dostopov do podatkov - transakcije (Transaction Control Language)

Uvod v SQL...



▪ Lastnosti SQL:

- Enostaven;
- Nepostopkoven (kaj in ne kako): enako ali bolj kot njegova teoretična osnova - relacijska algebra
- Uporaben v okviru številnih vlog: skrbniki PB, vodstvo, razvijalci informacijskih rešitev, končni uporabniki;
- ISO in ANSI standard;
- SQL je de-facto in tudi uradno standardni jezik za delo z relacijskimi podatkovnimi bazami.

Zgodovina SQL



- V 1970h IBM razvije sistem System R, ki temelji na relacijskem modelu Edgarja Codd-a.
- 1974 – D. Chamberlin in F. Boyce (IBM San Jose Laboratory) definirata jezik 'Structured English Query Language' (SEQUEL).
 - SEQUEL se kasneje preimenuje v SQL
- Pozno v 1970h – Relational Software (danes Oracle) razvije svoj SUPB, ki temelji na relacijskem modelu in implementira SQL.
- Poleti 1979 – Oracle izda prvo komercialno različico SQL; nekaj tednov pred IBM-ovo implementacijo za System/38

Standardizacija SQL



- 1986, SQL-86, ANSI izda prvo različico standarda. Leto kasneje potrdi še ISO (SQL-87).
- 1989, SQL-89, majhne revizije (FIPS 127-1).
- 1992, SQL-92 ali SQL2, večja revizija standarda.
- 1999, SQL:1999 ali SQL3, številne novosti: rekurzivne poizvedbe, prožilci, podpora proceduralni kodi, nekateri objektni dodatki...
- 2003, SQL:2003, dodane lastnosti za delo z XML, sekvence, avto-generiranje vrednosti...

Standardizacija SQL



- 2006, SQL:2006, ISO/IEC 9075-14:2006 definira način povezovanja SQL in XML; 2008 manjše dopolnitve
- 2011, SQL:2011, podpora časovnim podatkovnim bazam (podatkovna skladišča);
 - limited fetch: FETCH FIRST n ROWS, FETCH NEXT n ROWS
samo nekaj vrstic poizvedbe (doslej nestandardno: LIMIT, TOP, ...)
 - DML znotraj SELECT stavka
 - klici shranjenih procedur z imenovanimi in privzetimi argumenti
- 2016, SQL:2016, JSON, vzorci, polimorfne funkcije
 - JSON: kreiranje in dostop do JSON podatkovnega tipa
 - regularni izrazi nad več vrsticami
 - agregacija skupin v razmejene nize znakov
- 2019, SQL:2019, večdimenzionalna polja in operatorji

Dodatki za proceduralnost (od SQL:2003)



Vir	Naziv	Polno ime
ANSI	SQL/PSM	SQL/Persistent Stored Module
IBM	SQL PL	SQL Procedural Language
Microsoft/Sybase	T-SQL	Transact-SQL
MySQL	MySQL	MySQL
Oracle	PL/SQL	Procedural Language/SQL
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL

Proceduralnost (IF, LOOP, ...) je dosegljiva tudi prek integracije SUPB in splošno-namenskih programskih jezikov. Npr. SQL standard definira dodatek SQL/JRT za podporo Javi v relacijskih bazah. Ali pa dodatki za uporabo proceduralnih jezikov, npr. PostgreSQL: Python, Perl, Tcl, Lua, Java, R, ...

Pomembnost jezika SQL...



- SQL do sedaj **edini** široko sprejet standardni podatkovni jezik
 - ISO, ANSI
 - IBM - Systems Application Architecture (SAA)
 - Federal Information Processing Standard (FIPS) – standard kateremu morajo ustrezati vsi SUPB-ji prodani državnim organom v ZDA
 - ISO Information Resource Dictionary System (IRDS)
 - Remote Data Access (RDA)

- Paradoksalno: tudi uporaba v nerelacijskih (NoSQL) bazah kot plast nad nerelacijskim delom

Pomembnost jezika SQL



- Interesi v akademskih krogih so dali jeziku trdno teoretično osnovo
 - Optimizacija poizvedb
 - Porazdeljevanje podatkov
 - Varnost podatkov
- Specializirane implementacije SQL za analitiko (npr. Microsoft MDX, standardizirano od SQL:2019)
- Standardi v praksi: semantika enaka, razlike v detajlih in razširitvah, nihče ne podpira celotnih standardov
- Prenos SQL kode med sistemi je mogoč, vendar z manjšimi popravki



Pisanje SQL stavkov...

- SQL stavki so sestavljeni iz rezerviranih in uporabniško definiranih besed.
- Rezervirane besede so natančno določene, napisane morajo biti pravilno, ne smejo se lomiti med vrstice.
- Uporabniško definirane besede označujejo razne podatkovne objekte, kot so npr. relacije, stolpci, pogledi,...

Pisanje SQL stavkov...



- Po standardu je večina komponent SQL stavkov neodvisna od velikosti pisave; izjema so tekstovni podatki.
 - Izjeme, npr. MySQL na Linuxu (privzeta nastavitve imen tabel in atributov)
- Da dosežemo boljšo berljivost, pišemo SQL stavke v več vrsticah in z zamiki:
 - Vsak sklop SQL stavka se začne v novi vrstici
 - Sklopi so levo poravnani
 - Če ima sklop več delov, je vsak v svoji vrstici in poravnan z začetkom sklopa

Pisanje SQL stavkov...



- Za opis sintakse SQL stavkov bomo uporabljali naslednjo notacijo:
 - REZERVIRANE BESEDE z velikimi črkami,
 - uporabniško definirane besede z malimi črkami,
 - Znak | za izbiro med alternativami,
 - {Obvezni elementi} v zavutih oklepajih,
 - [Opcijski elementi] v oglatih oklepajih,
 - Znak ... za opsijske ponovitve (0 ali več).

Pisanje SQL stavkov



- Podatkovne vrednosti predstavljajo konstante v SQL stavkih.
- Vse ne-numerične vrednosti so zapisane v enojnih (ali dvojnih) narekovajih
'Ljubljana, ' "Janez Novak"
- Vse numerične vrednosti so brez narekovajev
225.990



Stavki skupine SQL DML



- Osnovna struktura za hranjenje podatkov je dvodimenzionalna tabela
- DML skupina zajema SQL stavke za manipulacijo s podatki v tabelah (in drugih strukturah)
 - SELECT → Izbira (branje podatkov iz PB)
 - INSERT → Dodajanje
 - DELETE → Brisanje
 - UPDATE → Spreminjanje
 - ... in še nekateri izvedeni (TRUNCATE, MERGE, ...)
- Sintaksa SELECT stavka najbolj kompleksna

Tabele v SQL



- Dvodimenzionalna struktura: vrstice × stolpci
- Stolpci (atributi)
 - poimenovani, vrstni red ni pomemben
 - imajo določen podatkovni tip
- Vrstice predstavljajo (pomensko ekvivalentno)
 - elemente neke množice objektov
 - opise objektov z atributi
 - entitete

Primeri tabel

Jadralec(jid, ime, rating, starost)

Coln(cid, ime, dolzina, barva)

Rezervacija(jid, cid, dan)

cid	ime	dolzina	barva
101	Elan	34	modra
102	Elan	34	rdeca
103	Sun Odyssey	37	zelena
104	Bavaria	50	rdeca

jid	cid	dan
22	101	10/10/2006
22	102	10/10/2006
22	103	10/8/2006
22	104	10/7/2006
31	102	11/10/2006
31	103	11/6/2006
31	104	11/12/2006
64	101	9/5/2006
64	102	9/8/2006
74	103	9/8/2006

jid	ime	rating	starost
22	Darko	7	45
29	Borut	1	33
31	Lojze	8	55.5
32	Andrej	8	25.5
58	Rajko	10	35
64	Henrik	7	35
71	Zdravko	10	16
74	Henrik	9	35
85	Anze	3	25.5
95	Bine	3	63.5

Podatkovni tipi
oz.
domene?

Stavek SELECT



```
SELECT [DISTINCT | ALL] { * | [columnExpression [AS newName]] [, ...] }  
FROM   TableName [alias] [, ...]  
[WHERE condition]  
[GROUP BY columnList]  
[HAVING condition]  
[ORDER BY columnList]
```


Preprosti stavki SELECT



```
-- Komentar preprostega stavka SELECT
SELECT A1, A2, ..., Ak    -- stolpci
FROM T1, T2, ..., Tn     -- ena ali več tabel
WHERE Pogoji;            -- filter vrstic
```

- Rezultat stavka **SELECT** kot začasna tabela
- Vrstice so elementi kartezičnega produkta tabel
 - **Pogoj** določa, katere izmed vrstic pridejo v rezultat
- **SELECT DISTINCT** ali **SELECT [ALL]** :
DISTINCT izloči duplikate iz rezultata;
privzeta vrednost **ALL** jih ohrani!



Stavek SELECT ...



- SELECT Določa stolpce, ki naj se pojavijo v izhodni relaciji
- FROM Določa tabele za poizvedbo
- WHERE Filtrira vrstice
- GROUP BY Združuje vrstice po vrednostih izbranih stolpcev
- HAVING Filtrira skupine glede na določene pogoje
- ORDER BY Določa vrstni red vrstic na izhodu

Vrstnega reda sklopov ni možno spreminjati!
Obvezna sta samo sklopa SELECT in FROM!

PostgreSQL: opis stavka
SELECT na 23 straneh!

Primeri SQL

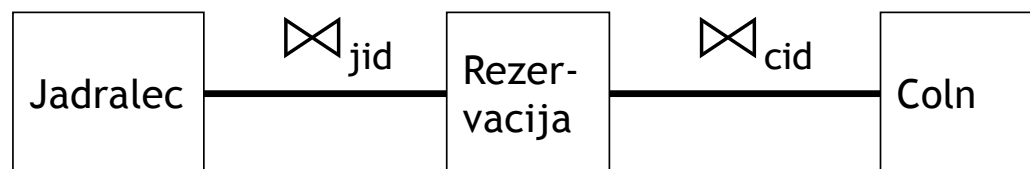
- Struktura tabel za primere

Jadralec(jid, ime, rating, starost)

Coln(cid, ime, dolzina, barva)

Rezervacija(#jid, #cid, dan)

- Podatkovne tipe zaenkrat zanemarimo
- Povezave med tabelami preko atributov:



Primeri

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)



- Izpiši vse podatke jadralcih:
 SELECT jid, ime, rating, starost
 FROM jadralec;
ali krajše
 SELECT jadralec.*
 FROM jadralec;
ali še krajše
 SELECT *
 FROM jadralec;

jid	ime	rating	starost
22	Darko	7	45
29	Borut	1	33
31	Lojze	8	55.5
32	Andrej	8	25.5
58	Rajko	10	35
64	Henrik	7	35
71	Zdravko	10	16
74	Henrik	9	35
85	Anze	3	25.5
95	Bine	3	63.5

Uporaba DISTINCT

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

- Izpiši oznake jadralcev, ki so kadarkoli rezervirali kakšen čoln

SELECT DISTINCT jid
FROM rezervacija;

Določilo DISTINCT odstrani podvojene vrstice iz rezultata poizvedbe.

+-----+
jid
+-----+
22
31
64
74
+-----+

Brez DISTINCT:
ostanejo
podvojene
vrstice.

Zakaj jih sploh
dobimo?

+-----+
jid
+-----+
22
64
22
31
64
22
31
74
22
31
+-----+

Jadralec(jid, ime, rating, starost)
 Coln(cid, ime, dolzina, barva)
 Rezervacija(jid, cid, dan)

Izračunana polja

- Izpiši desetkratnik ratinga vsakega jadralca.

Izračunanemu stolpcu
dodelimo naziv

```
SELECT jid, ime, rating*10 AS Desetkratnik
FROM jadralec;
```

Uporabljamo formule

+-----+-----+-----+-----+			
jid	ime		Desetkratnik
+-----+-----+-----+-----+			
22	Darko		70
29	Borut		10
31	Lojze		80
32	Andrej		80
58	Rajko		100
64	Henrik		70
71	Zdravko		100
74	Henrik		90
85	Anze		30
95	Bine		30
+-----+-----+-----+-----+			

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

Iskalni kriteriji

- Izpiši oznake rdečih čolnov (barva = 'rdeca') in dolžino manjšo kot 40 čevljev.

```
SELECT cid  
FROM coln  
WHERE barva='rdeca' AND dolzina < 40;
```

Pogoj 1

Pogoj 2

Pogoje združujemo z logičnimi operatorji

```
+-----+  
| cid |  
+-----+  
| 102 |  
+-----+
```

Iskanje z uporabo BETWEEN

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

- Izpiši šifre in imena jadralcev starih med 35 in 45 let.

```
SELECT jid, ime, starost
```

```
FROM jadralec
```

```
WHERE starost BETWEEN 35 AND 45;
```

BETWEEN vključuje spodnjo in zgornjo mejo.

Uporabimo lahko tudi negacijo NOT BETWEEN.

BETWEEN nima dodatne izrazne moči, izraz

“x BETWEEN sp AND zg” je možno izraziti

posredno:

sp <= x AND x <= zg

+-----+-----+-----+		
jid	ime	starost
+-----+-----+-----+		
22	Darko	45
58	Rajko	35
64	Henrik	35
74	Henrik	35
+-----+-----+-----+		

Iskanje po članstvu množice (\in , \notin)

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

- Izpiši oznake in imena čolnov rdeče ali zelene barve.

```
SELECT cid, ime
```

```
FROM coln
```

```
WHERE barva='rdeca'
```

```
      OR barva='zelena';
```

```
-- ali
```

```
WHERE barva IN ('rdeca', 'zelena');
```

+-----+-----+-----+	
cid	ime
+-----+-----+-----+	
102	Elan
103	Sun Odyssey
104	Bavaria
+-----+-----+-----+	

← Članstvo množice

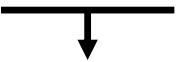
Uporabimo lahko tudi negacijo NOT IN. V takšni obliki IN ne doda veliko izrazne moči; koristen pa je pri večjih dinamičnih množicah

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

Iskanje z vzorcem

- Izpiši imena vseh jadralcev, ki vsebujejo črko 'j'.

```
SELECT ime  
FROM jadralec  
WHERE ime LIKE '%j%';
```


Iskanje z vzorcem

ime
Lojze
Andrej
Rajko

SQL ima dva posebna znaka za **preprosto** iskanje z vzorcem:

Znak % nadomešča katerikoli niz znakov

Znak _ nadomešča katerikoli znak

SQL: regularni izrazi

Jadralec(jid, ime, rating, starost)
 Coln(cid, ime, dolzina, barva)
 Rezervacija(jid, cid, dan)

- Izpiši podatke o jadralcih, katerih imena se začnejo na B in imajo najmanj 5 črk.

SELECT *

FROM jadralec

WHERE ime REGEXP '^B.....*\$';

-- MySQL

-- ali (REGEXP = RLIKE)

WHERE ime RLIKE '^B[a-z]{4}[a-z]*\$';

-- MySQL

-- ali

WHERE ime ~ '^B[a-z]{4}[a-z]*\$';

-- PostgreSQL

-- Posix

-- sintaksa

-- regularnih

-- izrazov

jid	ime	rating	starost
29	Borut	1	33

SQL: regularni izrazi



Kaj pomeni: '^B[a-z]{4}[a-z]*\$'

- Specifikacija vzorca s Posix regularnim izrazom
- ^ označuje začetek, \$ pa konec niza (sicer se išče poljuben podniz)
- . (pika) ustreza natanko eni poljubnemu znaku
- | pomeni alternativo [VELIK|majhen]
- [a-z] je katera koli črka z intervala med a in z, ekvivalent alternativam [a|b|c|...|w|x|y|z]
- * pomeni 0 ali več ponovitev predhodnega znaka
- + pomeni 1 ali več ponovitev predhodnega znaka

SQL: standardizirani regularni izrazi



SELECT *	-- SQL:1999
FROM jadralec	-- sintaksa
WHERE ime SIMILAR TO 'B[a-z]{4}[a-z]*';	-- (PostgreSQL)

- SQL:1999: standardni operator SIMILAR TO za delo z regularnimi izrazi
- Redko implementirano, npr. PostgreSQL
- Sintaksa regularnih izrazov: križanec med LIKE in Posix
 - `_` namesto `.`, `%` sinonim za `*`
 - ne pozna `^` in `$`, predpostavlja implicitno uporabo; če tega nočemo, dodamo `_*` na začetek in konec vzorca:

WHERE ime SIMILAR TO ' `_*` B[a-z]{4}[a-z]* `_*` ';

Iskanje z oznako NULL v pogoju

- NULL: posebna oznaka (ne vrednost!), ki označuje nedefiniran ali manjkajoč podatek
- Izpiši vse jadralce brez ratinga

```
SELECT jid  
FROM jadralec  
WHERE rating IS NULL;
```

Iskanje z NULL oznako

Uporabljamo lahko
tudi negacijo IS
NOT NULL

```
+-----+  
| jid |  
+-----+  
+-----+
```

SQL: trivrednostna logika (T, F, ?)

p OP q =		p		
		V1	V2	NULL
q	V1	True	False	NULL
	V2	False	True	NULL
	NULL	NULL	NULL	NULL

Če sta oba operanda definirana (torej NOT NULL) poteka primerjava kot običajno, sicer pa je rezultat neznan (NULL).

(NULL=NULL) IS NULL ???

Podobno velja za ostale podatkovne tipe in operatorje (<, >, >=, <=, +, -, *, /, ...)

Če pričakujemo neznane (NULL) oznake, moramo to upoštevati z uporabo dodatnih preverjanj IS NULL, IS NOT NULL

SQL: trivrednostna logika (T, F, ?), logični izrazi

p AND q				
		True	False	NULL
q	True	True	False	NULL
	False	False	False	False
	NULL	NULL	False	NULL

p OR q		p		
		True	False	NULL
q	True	True	True	True
	False	True	False	NULL
	NULL	True	NULL	NULL

p	NOT p
True	False
False	True
NULL	NULL

p in q sta logična izraza

NULL se lahko pojavi tudi kot rezultat poljubnega logičnega ali aritmetičnega izraza, npr.
(NULL > 10) ali (NULL*10)

Urejanje vrstic v rezultatu

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)



- Izpiši vse podatke o jadralcih, urejene po starosti od najmlajšega do najstarejšega, znotraj enake starosti pa po ratingu od največjega do najmanjšega.

```
SELECT *  
FROM jadralec  
ORDER BY starost ASC,  
         rating DESC;
```

ASC - ascending → naraščujoče
(privzeta vrednost)

DESC - descending → padajoče

jid	ime	rating	starost
71	Zdravko	10	16
32	Andrej	8	25.5
85	Anze	3	25.5
29	Borut	1	33
58	Rajko	10	35
74	Henrik	9	35
64	Henrik	7	35
22	Darko	7	45
31	Lojze	8	55.5
95	Bine	3	63.5

Poizvedbe po več tabelah (neformalno)



- Poizvedovanje po podatkih dobi pravo moč šele, ko združimo podatke iz več tabel
⇒ poizvedovanje istočasno po več tabelah
- Vrstice med tabelami na nek način povežemo
- Podmnožica kartezičnega produkta
 - kartezični produkt:
vsak element prve množice (vrstico prve tabele)
združimo z
vsakim elementom druge množice (vrstico druge tabele)
 - samo nekatere sestavljene vrstice so zanimive (podmnožica)
- Definirajmo: stik (join) kot "zanimiva" podmnožica kartezičnega produkta

Poizvedbe po več tabelah...



- Kartezični produkt izvedemo (po definiciji) tako, da v sklopu FROM navedemo imena tabel
- V sklopu WHERE pa določimo logični pogoj za podmnožico "zanimivih" vrstic
- Če v tabelah nastopajo atributi z enakim imenom, jih moramo opredeliti še z imeni tabel, ki jim pripadajo.

```
SELECT jadralec.ime, coln.ime  
FROM jadralec j, coln c  
WHERE ... ;
```

Poizvedbe po več tabelah...



- Za tabele v razdelku FROM lahko uvedemo sinonime (alias), ki olajšajo poimenovanje.
- S sinonimom začasno preimenujemo tabelo!
- Sintaksa:

```
SELECT j.ime, c.ime  
FROM jadralec j, coln c  
WHERE ...
```
- Sinonimi so potrebni za ločevanje med istoimenskimi stolpci različnih tabel ali imeni stolpcev "kopij" iste tabele.

Primer poizvedbe po dveh tabelah

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

- Izpiši podatke o rezerviranih zelenih čolnih


```
SELECT *  
FROM coln, rezervacija  
WHERE barva='zelena'; → Samo zeleni čolni
```

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

Primer poizvedbe po dveh tabelah

Jadralec(jid, ime, rating, starost)
Coln(cid, ime, dolzina, barva)
Rezervacija(jid, cid, dan)

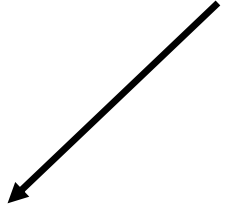
- Izpiši podatke o rezerviranih zelenih čolnih

SELECT c.*  Vsi atributi specificirane tabele.
Samo *: vsi atributi vseh tabel.

FROM coln c, rezervacija r

WHERE c.cid = r.cid AND c.barva='zelena';  Samo zeleni čolni

Samo
vrstice,
ki se
ujemajo
v cid



cid	ime	dolzina	barva
103	Sun Odyssey	37	zelena
103	Sun Odyssey	37	zelena
103	Sun Odyssey	37	zelena

Primer poizvedbe po več tabelah

- Izpiši oznake in imena jadralcev, ki so rezervirali rdeče čolne na dan 10. 10. 2006.

```
SELECT j.jid, j.ime  
FROM jadralec j, rezervacija r, coln c  
WHERE  j.jid=r.jid AND r.cid=c.cid  
      AND c.barva='rdeca'  
      AND r.dan=DATE'2006-10-10';
```

+	-----	+	-----	+
	jid		ime	
+	-----	+	-----	+
	22		Darko	
+	-----	+	-----	+

Datumska
konstanta
(literal)

Semantika poizvedovanja po več tabelah



- Potrebujemo dodatno teoretično znanje.

Podatkovni modeli



Relacijski podatkovni model



Relacijska algebra in relacijski račun



Povporaševalni jezik SQL