Sadržaj

Predgovor ..... **xi**

1. Preuzimanje zapisa **1**

1.1 Preuzimanje svih redova i kolona iz tabele 1
1.2 Preuzimanje podskupa redova iz tabele 2
1.3 Pronalaženje redova koji zadovoljavaju više uslova 2
1.4 Preuzimanje podskupa kolona iz tabele 3
1.5 Davanje smislenih imena za kolone 3
1.6 Referenciranje kolone sa alijasom u klauzuli WHERE 4
1.7 Nadovezivanje vrednosti kolona 5
1.8 Korišćenje uslovne logike u SELECT izjavi 7
1.9 Ograničavanje broja vraćenih redova 8
1.10 Vraćanje n nasumičnih zapisa iz tabele 9
1.11 Pronalaženje null vrednosti 11
1.12 Pretvaranje null u stvarne vrednosti 11
1.13 Traženje obrazaca 12
1.14 Rezime 13

2. Sortiranje rezultata upita **15**

2.1 Vraćanje rezultata upita po željenom redosledu 15
2.2 Sortiranje po više polja 16
2.3 Sortiranje po podznakovnim nizovima 17
2.4 Sortiranje mešovitih alfanumeričkih podataka 18
2.5 Rad sa NULL pri sortiranju 21
2.6 Sortiranje po ključu zavisnom od podataka 27
2.7 Rezime 28

3. Rad sa više tabela **29**

3.1 Slaganje jednog skupa redova na drugi 29
3.2 Kombinovanje srodnih redova 31
3.3 Pronalaženje zajedničkih redova za dve tabele 32
3.4 Preuzimanje vrednosti iz jedne tabele koje ne postoje u drugoj 34
3.5 Preuzimanje redova iz jedne tabele koji se ne podudaraju sa redovima u drugoj 40
3.6 Dodavanje spojeva upitu bez ometanja drugih spojeva 41

3.7 Utvrđivanje da li dve tabele imaju iste podatke 43
3.8 Identifikovanje i izbegavanje kartezijanskih proizvoda 50
3.9 Izvođenje spajanja kada se koristi grupisanje 52
3.10 Izvođenje spoljašnjih spajanja kada se koristi grupisanje 56
3.11 Vraćanje nedostajućih podataka iz više tabela 59
3.12 Korišćenje NULL u operacijama i poređenjima 63
3.13 Rezime 64

4. Umetanje, ažuriranje i brisanje 65
4.1 Umetanje novog zapisa 65
4.2 Umetanje podrazumevanih vrednosti 66
4.3 Zamena podrazumevane vrednosti sa NULL 68
4.4 Kopiranje redova iz jedne tabele u drugu 68
4.5 Kopiranje definicije tabele 69
4.6 Umetanje u više tabela odjednom 70
4.7 Blokiranje umetanja u određene kolone 72
4.8 Izmena zapisa u tabeli 73
4.9 Ažuriranje kada odgovarajući redovi postoje 74
4.10 Ažuriranje vrednostima iz druge tabele 75
4.11 Spajanje zapisa 78
4.12 Brisanje svih zapisa iz tabele 80
4.13 Brisanje određenih zapisa 80
4.14 Brisanje jednog zapisa 81
4.15 Brisanje narušavanja referentnog integriteta 81
4.16 Brisanje duplikata zapisa 82
4.17 Brisanje zapisa referisanih u drugoj tabeli 84
4.18 Rezime 85

5. Upiti za metapodatke 87
5.1 Prikazivanje tabela po šemi 87
5.2 Prikazivanje kolona tabele 88
5.3 Prikazivanje indeksiranih kolona tabele 89
5.4 Prikazivanje ograničenja za tabelu 91
5.5 Prikazivanje spoljašnjih ključeva bez odgovarajućih indeksa 92
5.6 Korišćenje SQL-a za generisanje SQL-a 95
5.7 Opis prikaza rečnika podataka u Oracle bazi podataka 97
5.8 Rezime 99

6. Rad sa znakovnim nizovima 101
6.1 Prolaženje kroz znakovni niz 101
6.2 Umetanje navodnika u znakovni niz 104
6.3 Brojanje pojava znakova u nizu znakova 105
6.4 Uklanjanje neželjenih znakova iz znakovnog niza 106
6.5 Odvajanje numeričkih i znakovnih podataka 107

6.6 Određivanje da li je niz alfanumerički 112
6.7 Izdvajanje inicijala iz imena 116
6.8 Ređanje po delovima znakovnog niza 120
6.9 Redosled po broju u znakovnom nizu 121
6.10 Pravljenje razgraničene liste iz redova tabele 127
6.11 Pretvaranje razgraničenih podataka u viševrednosnu IN-listu 130
6.12 Abecedni poredak u znakovnom nizu 136
6.13 Identifikovanje znakovnih nizova koji se mogu tretirati kao brojevi 141
6.14 Izdvajanje n-tog razgraničenog podniza 148
6.15 Raščlanjivanje IP adrese 154
6.16 Upoređivanje znakovnih nizova po zvučnosti 156
6.17 Pronalaženje teksta koji se ne podudara sa obrascem 158
6.18 Rezime 161

7. Rad sa brojevima....163

7.1 Izračunavanje proseka 163
7.2 Pronalaženje minimalne/maksimalne vrednosti u koloni 165
7.3 Sumiranje vrednosti u koloni 167
7.4 Brojanje redova u tabeli 169
7.5 Brojanje vrednosti u koloni 171
7.6 Generisanje kumulativnog zbira 171
7.7 Generisanje kumulativnog proizvoda 173
7.8 Glačanje niza vrednosti 174
7.9 Izračunavanje moda 175
7.10 Izračunavanje medijane 178
7.11 Određivanje procenta totala 180
7.12 Grupisanje kolona koje mogu sadržati NULL 183
7.13 Izračunavanje proseka bez visokih i niskih vrednosti 184
7.14 Pretvaranje alfanumeričkih znakovnih nizova u brojeve 186
7.15 Promena vrednosti u kumulativnog sumi 188
7.16 Pronalaženje anomalija pomoću medijane apsolutne devijacije 190
7.17 Pronalaženje anomalija pomoću Benfordovog zakona 194
7.18 Rezime 195

8. Aritmetika datuma....197

8.1 Sabiranje i oduzimanje dana, meseci i godina 197
8.2 Određivanje broja dana između dva datuma 200
8.3 Određivanje broja radnih dana između dva datuma 202
8.4 Određivanje broja meseci ili godina između dva datuma 207
8.5 Određivanje broja sekundi, minuta ili sati između dva datuma 209
8.6 Brojanje pojava dana sedmice u godini 211
8.7 Utvrđivanje razlike u datumu između tekućeg zapisa i narednog zapisa 223
8.8 Rezime 228

9. Manipulacija datumima 229
9.1 Utvrđivanje da li je godina prestupna 229
9.2 Određivanje broja dana u godini 236
9.3 Izdvajanje jedinica vremena iz datuma 239
9.4 Određivanje prvog i poslednjeg dana u mesecu 241
9.5 Određivanje svih datuma za određeni dan u sedmici tokom godine 244
9.6 Određivanje datuma prvog i poslednjeg pojavljivanja određenog dana sedmice u mesecu 250
9.7 Izrada kalendara 257
9.8 Spisak kvartalnih datuma početka i završetka godine 270
9.9 Određivanje datuma početka i kraja kvartala za dati kvartal 274
9.10 Popunjavanje nedostajućih datuma 281
9.11 Pretraživanje određenih vremenskih jedinica 288
9.12 Upoređivanje zapisa korišćenjem određenih delova datuma 290
9.13 Identifikovanje preklapajućih datumskih perioda 293
9.14 Rezime 298

10. Rad sa opsezima..... 299
10.1 Nalaženje niza uzastopnih vrednosti 299
10.2 Pronalaženje razlika između redova u istoj grupi ili particiji 302
10.3 Lociranje početka i kraja niza uzastopnih vrednosti 308
10.4 Popunjavanje nedostajućih vrednosti u opsegu vrednosti 311
10.5 Generisanje uzastopnih numeričkih vrednosti 315
10.6 Rezime 318

11. Napredno pretraživanje..... 319
11.1 Podela na stranice skupa rezultata 319
11.2 Preskakanje n redova tabele 321
11.3 Uključivanje OR logike pri korišćenju spoljnjih spojeva 323
11.4 Određivanje recipročnih redova 325
11.5 Izbor najboljih n zapisa 327
11.6 Pronalaženje zapisa sa najvišim i najnižim vrednostima 328
11.7 Istraživanje budućih redova 329
11.8 Promena vrednosti redova 331
11.9 Rangiranje rezultata 334
11.10 Suzbijanje duplikata 334
11.11 Pronalaženje vrednosti skakača 337
11.12 Generisanje jednostavnih predviđanja 342
11.13 Rezime 351

12. Izveštavanje i preoblikovanje 353
12.1 Okretanje skupa rezultata u jedan red 353
12.2 Okretanje skupa rezultata u više redova 356

12.3 Okretanje skupa rezultata unazad 361
12.4 Okretanje skupa rezultata u jednu kolonu unazad 363
12.5 Suzbijanje ponavljajućih vrednosti iz skupa rezultata 366
12.6 Okretanje skupa rezultata radi olakšavanja proračuna između redova 368
12.7 Stvaranje segmenata podataka fiksne veličine 370
12.8 Stvaranje unapred definisanog broja segmenata 372
12.9 Pravljenje horizontalnih histograma 373
12.10 Pravljenje vertikalnih histograma 375
12.11 Vraćanje kolona koje nisu u GROUP BY 377
12.12 Izračunavanje jednostavnih međusuma 380
12.13 Izračunavanje međusuma za sve moguće kombinacije izraza 384
12.14 Identifikovanje redova koji nisu međusume 394
12.15 Korišćenje izraza CASE za označavanje redova 396
12.16 Kreiranje proređene matrice 397
12.17 Grupisanje redova po jedinicama vremena 399
12.18 Istovremeno izvođenje grupisanja nad različitim grupama/participama 403
12.19 Izvođenje grupisanja na pokretnom opsegu vrednosti 404
12.20 Okretanje skupa rezultata sa međusumama 412
12.21 Rezime 417

13. Hijerarhijski upiti 419

13.1 Predstavljanje odnosa roditelja i deteta 420
13.2 Prikazivanje odnosa dete-roditelj-deda 423
13.3 Kreiranje hijerarhijskog prikaza tabele 428
13.4 Pronalaženje svih redova dece za dati red roditelja 432
13.5 Utvrđivanje koji su redovi list, grana ili korenski čvorovi 433
13.6 Rezime 441

14. Razne male stvari 443

14.1 Unakrsni izveštaji pomoću operatora PIVOT SQL Servera 443
14.2 Poništavanje okretanja unakrsnih izveštaja pomoću UNPIVOT operatora SQL Servera 445
14.3 Preoblikovanje skupa rezultata pomoću Oracleove klauzule MODEL 447
14.4 Izdvajanje elemenata znakovnog niza iz lokacija koje nisu fiksne 451
14.5 Pronalaženje broja dana u godini (alternativno rešenje za Oracle) 454
14.6 Pretraživanje mešovitih alfanumeričkih znakovnih nizova 455
14.7 Pretvaranje celih brojeva u binarne pomoću Oraclea 457
14.8 Okretanje rangiranog skupa rezultata 460
14.9 Dodavanje zaglavlja kolone u dvostruko obrnutom skupu rezultata 464
14.10 Pretvaranje skalarnog podupita u kompozitni podupit u Oracleu 476
14.11 Raščlanjivanje serializovanih podataka u redove 478
14.12 Izračunavanje procenta u odnosu na ukupan iznos 483
14.13 Testiranje postojanja vrednosti unutar grupe 485
14.14 Rezime 488

A. Potsetnik za funkcije prozora 489

B. Uobičajeni izrazi tabela 515

Indeks 519

Predgovor

SQL je zajednički jezik stručnjaka za obradu podataka. Istovremeno, ne dobija uvek pažnju koju zaslužuje u poređenju sa ostalim glavnim alatima. Kao rezultat toga, uobičajeno je pronaći ljude koji često koriste SQL, ali retko ili nikad ne idu dalje od najjednostavnijih upita, često zato što veruju da ono što znaju je sve što i postoji.

Ova knjiga pokazuje koliko mnogo SQL može učiniti, proširujući vašu kutiju alatki. Na kraju knjige videćete kako se SQL može koristiti za statističke analize; da pravi izveštaje na sličan način kao što su profesionalni poslovni alati; za podudaranje tekstualnih podataka; za sofisticiranu analizu datumskih podataka; i još mnogo toga.

Prvo izdanje knjige *SQL kuvar*, od svog izdavanja, bilo je popularan izbor kao „druga knjiga o SQL-u" – knjiga koju ljudi rado čitaju nakon što nauče osnove. Ima mnoge kvalitete, poput širokog spektra tema i čitljivog stila pisanja.

Poznato je da se računarstvo brzo razvija, čak i kada je reč o nečem tako zrelom kao što je SQL, koji vuče korene još iz 1970-ih. Iako ovo novo izdanje ne sadrži potpuno nove jezičke mogućnosti, važna promena je što su funkcije koje su bile nove u vreme prvog izdanja sada stabilizovane i standardizovane. Kao rezultat, imamo mnogo više prostora za razvoj standardnih rešenja nego što je to bilo moguće ranije.

Dva su ključna primera koje je važno naglasiti. Uobičajeni izrazi tabela (eng. *common table expressions, CTE*), uključujući rekurzivne CTE, bili su dostupni u nekoliko implemenatcija u vreme objavljivanja prvog izdanja, ali su sada dostupni u svih pet. Uvedeni su kako bi rešili neka praktična ograničenja SQL-a, od kojih se neka mogu videti direktno u ovim receptima. Novi dodatak o rekurzivnim CTE u ovom izdanju naglašava njihov značaj i objašnjava njihovu važnost.

Funkcije prozora (eng. *window functions*) su takođe bile prilično nove u vreme izlaska prvog izdanja i nisu bile dostupne u svakoj implementacij. Takođe su bile dovoljno nove da je napisan poseban dodatak koji ih objašnjava. Sada su funkcije prozora u svim implementacijama koje ova knjiga obuhvata. One su i u svim ostalim implementacijama SQL-a koje su nam poznate, mada postoji toliko mnogo baza podataka, da je nemoguće garantovati da sve imaju funkcije prozora i/ili CTE.

Pored standardizacije upita gde god je to moguće, uneli smo novi materijal u poglavlja 6 i 7. Materijal u poglavlju 7 donose nove aplikacije za analizu podataka u receptima o srednjem apsolutnom odstupanju i Benfordovom zakonu. U poglavlju 6 imamo novi recept koji pomaže u poklapanju podataka prema zvučnosti, a materijal o regularnim izrazima prebacili smo iz poglavlja 14 u poglavlje 6.

Kome je namenjena ova knjiga

Ova knjiga je namenjena svakom SQL korisniku koji želi da dalje razvija svoje upite. Što se tiče zahteva, namenjeno je nekome ko poznaje barem neki SQL, a idealno bi bilo da znate da pišete upite na podacima sa stvarnim problemima iz prakse.

Ovo je knjiga za sve korisnike SQL-a, uključujući stručnake za podatake, ljude koji vizualizuju podatke itd. Neki od ovih korisnika možda nikada ili retko pristupaju bazama podataka direktno, ali vizualizuju svoje podatke ili preko statističkih alata postavljaju upite i preuzimaju podatake. Akcenat je na praktičnim upitima koji mogu rešiti probleme iz stvarnog sveta. Tamo gde se pojavljuje nešto teorije, ona je tu da direktno podrži praktične elemente.

Šta nedostaje u ovoj knjizi

Ovo je praktična knjiga, uglavnom o korišćenju SQL-a za razumevanje podataka. Ne sadrži teorijske aspekte baza podataka, dizajn baze podataka ili teoriju iza SQL-a, osim tamo gde je potrebno da bi se objasnili određeni recepti ili tehnike.

Takođe ne obuhvata proširenja baza podataka za obradu tipova podataka kao što su XML i JSON. Za te specijalističke teme postoje drugi resursi.

Platforma i verzija

SQL je pokretna meta. Proizvođači neprekidno dodaju nove funkcije i funkcionalnosti u svoje proizvode. Stoga trebalo bi znati koje verzije platformi su korišćene u pripremi teksta ove knjige:

- DB2 11.5
- Oracle baza podataka 19c
- PostgreSQL 12
- SQL Server 2017
- MySQL 8.0

Tabele korišćene u ovoj knjizi

Većina primera u ovoj knjizi uključuje upotrebu dve tabele, EMP i DEPT. EMP tabela, tabela zaposlenih (eng. *employee*), je jednostavna tabela od 14 redova koja sadrži samo numerička, znakovna i datumska polja. Tabela DEPT, tabela odeljenja (eng. *department*), je jednostavna tabela od 4 reda koja sadrži samo numerička i znakovna polja. Ove tabele se pojavljuju u mnogim starim udžbenicima za baze podataka, a odnos „više prema jedan" između odeljenja i zaposlenih je jasno razumljiv.

U ovoj knjizi, skoro sva rešenja koriste ove tabele. Nigde ne doterujemo primere podataka da bismo dobili rešenje koje verovatno ne biste imali priliku da primenite u stvarnom svetu, kao što je to slučaj u nekim knjigama.

Ovde su prikazani sadržaji EMP i DEPT:

```
 select * from emp;

EMPNO ENAME  JOB        MGR HIREDATE     SAL COMM  DEPTNO
----- ------ --------- ---- ----------- ---- ---- -------
 7369 SMITH  CLERK     7902 17-DEC-2005  800           20
 7499 ALLEN  SALESMAN  7698 20-FEB-2006 1600  300      30
 7521 WARD   SALESMAN  7698 22-FEB-2006 1250  500      30
 7566 JONES  MANAGER   7839 02-APR-2006 2975           20
 7654 MARTIN SALESMAN  7698 28-SEP-2006 1250 1400      30
 7698 BLAKE  MANAGER   7839 01-MAY-2006 2850           30
 7782 CLARK  MANAGER   7839 09-JUN-2006 2450           10
 7788 SCOTT  ANALYST   7566 09-DEC-2007 3000           20
 7839 KING   PRESIDENT      17-NOV-2006 5000           10
 7844 TURNER SALESMAN  7698 08-SEP-2006 1500    0      30
 7876 ADAMS  CLERK     7788 12-JAN-2008 1100           20
 7900 JAMES  CLERK     7698 03-DEC-2006  950           30
 7902 FORD   ANALYST   7566 03-DEC-2006 3000           20
 7934 MILLER CLERK     7782 23-JAN-2007 1300           10


  select * from dept;

DEPTNO DNAME          LOC
------ -------------- ---------
     10 ACCOUNTING     NEW YORK
     20 RESEARCH       DALLAS
     30 SALES          CHICAGO
     40 OPERATIONS     BOSTON
```

Pored toga, u ovoj knjizi naći ćete četiri izvedene (pivot) tabele: T1, T10, T100 i T500. Budući da ove tabele postoje samo da bi olakšale izvođenje, nismo im dali smislenija imena. Broj koji sledi „T" u svakoj od ovih tabela označava broj redova u svakoj tabeli, počev od 1. Na primer, ovde su vrednosti za T1 i T10:

```
select id from t1;

        ID
----------
         1

select id from t10;

        ID
----------
         1
```

```
 2
 3
 4
 5
 6
 7
 8
 9
10
```

Pivot tabele su korisna prečica kada treba da napravimo niz redova kako bismo olakšali upit.

Osim toga, neki dobavljači dozvoljavaju delimične SELECT izraze. Na primer, možete imati SELECT bez klauzule FROM. Ponekad ćemo u ovoj knjizi koristiti tablicu podrške, T1, sa jednim redom, umesto da koristimo delimične upite, zarad jasnoće. Ovo je slično korišćenju Oracleove DUAL tabele, ali koristeći T1 tabelu radimo istu stvar na standardizovan način u svim implementacijama koje pokrivamo.

Sve druge tabele su specifične za određene recepte i poglavlja i biće uvedene u tekst kada je to potrebno.

Konvencije korišćene u ovoj knjizi

U ovoj knjizi koristimo brojne tipografske konvencije i konvencije kodiranja. Odvojite trenutak vremena da se upoznate sa njima. To će poboljšati razumevanje teksta. Konvencije kodiranja su posebno važne, jer ih ne možemo ponoviti u svakom receptu u knjizi. Umesto toga, ovde navodimo važne konvencije.

Tipografske konvencije

U ovoj knjizi se koriste sledeći tipografski dogovori:

VELIKA SLOVA
: Koristi se za označavanje rezervisanih reči SQL-a

mala slova
: Koristi se za sve upite u primerima koda. Ostali jezici kao što su C i Java koriste mala slova za većinu rezervisanih reči i smatramo ih daleko čitljivijima od velikih slova. Stoga će svi upiti biti napisani malim slovima.

`Fiksna širina podebljana`
: Označava unos korisnika u primerima sa interakcijom.


Označava savet, predlog ili opštu napomenu.



Označava upozorenje ili oprez.

Konvencije o kodiranju

Naš prioritet u odnosu na velika i mala slova u SQL izrazima je da uvek koristimo mala slova, kako za rezervisane reči, tako i za korisničke identifikatore. Na primer:

```
select empno, ename
  from emp;
```

Vaši prioriteti mogu biti drugačiji. Na primer, mnogi više vole da pišu velikim slovima rezervisane reči SQL-a. Koristite bilo koji stil kodiranja koji želite ili šta god vaš projekat zahteva.

Uprkos upotrebi malih slova u primerima koda, u tekstu dosledno koristimo velika slova za rezervisane reči i identifikatore SQL-a. Činimo to da bi se te stavke istakle u odnosu na okolni tekst. Na primer:

> Prethodni upit predstavlja upotrebu SELECT na tabeli EMP.

Iako ova knjiga obuhvata baze podataka pet različitih dobavljača, odlučili smo da koristimo jedan format za sve izlaze:

```
 EMPNO ENAME
----- ------
  7369 SMITH
  7499 ALLEN
```

Mnoga rešenja koriste *umetnute prikaze* (eng. *inline view,*) ili podupite u klauzuli FROM. ANSI SQL standard zahteva da se takvim prikazima daju pseudonimi, tj. alijasi tabele. (Oracle je jedini dobavljač koji vam omogućava da ne koristite alijase.) Dakle, naša rešenja koriste alijase kao što su X i Y za identifikaciju skupova rezultata iz umetnutih prikaza:

```
 select job, sal
from (select job, max(sal) sal
        from emp
      group by job)x;
```

Obratite pažnju na slovo X nakon završne, zatvarajuće zagrade. To slovo X postaje ime „tabele" koju vraća podupit u klauzuli FROM. Iako su alijasi kolona dragocen alat za pisanje samodokumentujućeg koda, alijasi umetnutih prikaza (za većinu recepata u ovoj knjizi) su samo formalnosti. Obično im se daju trivijalna imena kao što su X, Y, Z, TMP1 i TMP2. U slučajevima kada bolji alijas može pružiti bolje razumevanja, koristimo ih.

Videćete da je SQL dat u odeljku „Rešenje" recepata obično numerisan, na primer:

```
1 select ename
2     from emp
3  where deptno = 10
```

Broj nije deo sintakse; to je samo upućivanje na delove upita kasnije u odeljku „Objašnjenje".

Mikro knjiga

Preko 37 godina Mikro knjiga donosi odabrane IT naslove i naslove opšteobrazaovnih tema u ediciji *Za neupućene*... Sa preko 460 objavljenih naslova pomogla je u obrazovanju desetina hiljada IT stručnjaka. Za više informacija o našim izdanjima pogledaje na *www.mikroknjiga.rs*

Kako da nas kontaktirate

MIKRO KNJIGA DOO
Kneza Višeslava 34
11030 Beograd
+381 11 3055010, 3055015

e-mail: *redakcija@mikroknjiga.rs*
Facebook: *http://facebook.com/mikroknjiga*
Twitter: *http://twitter.com/mikroknjiga*

Veb prikaz ove knjige, gde su, dodatne informacije i primeri, nalazi se na adresi *https://www.mikroknjiga.rs/store/prikaz.php?ref=SQ2K*

Sve vaše sugestije, primedbe i prijave eventualnih grešaka u knjizi su dobrodošle.

POGLAVLJE 1

Preuzimanje zapisa

Ovo poglavlje se fokusira na osnovne SELECT izjave. Važno je dobro razumeti osnove, jer su mnoge teme ovde obrađene ne nalaze samo u težim receptima već i u svakodnevnom SQL-u.

1.1 Preuzimanje svih redova i kolona iz tabele

Zadatak

Imate tabelu i želite da vidite sve podatke u njoj.

Rešenje

Koristite specijalni * znak i zadajte SELECT na tabeli:

```
1 select *
2   from emp
```

Objašnjenje

Znak * ima posebno značenje u SQL-u. Njegova upotreba će vratiti sve kolone navedene tabele. Budući da nije navedena klauzula WHERE, vratiće se i svaki red. Alternativa bi bila da se svaka kolona navede pojedinačno:

```
select empno,ename,job,sal,mgr,hiredate,comm,deptno
  from emp
```

U kratkim upitima koje izvršavate interaktivno lakše je koristiti SELECT *. Međutim, prilikom pisanja programskog koda, bolje je navesti svaku kolonu pojedinačno. Učinak će biti isti, ali ako budete izričiti, uvek ćete znati koje kolone vraćate iz upita. Isto tako, takve upite lakše razumeju ljudi koji nisu vi (koji možda znaju ili ne znaju sve kolone u tabelama u upitu). Problemi sa SELECT * mogu se pojaviti i ako je vaš upit unutar koda, a program iz upita dobija drugačiji skup kolona nego što se očekivalo. Ako navedete sve kolone, a jedna ili više nedostaje, veća je verovatnoća da će se svaka greška voditi do određenih kolona koje nedostaju.

1.2 Preuzimanje podskupa redova iz tabele

Zadatak

Imate tabelu i želite da vidite samo redove koji zadovoljavaju određeni uslov.

Rešenje

Koristite WHERE klauzulu da zadate koje redove treba zadržati. Na primer, za pregled svih zaposlenih raspoređenih u odeljenje broj 10:

```
1 select *
2   from emp
3  where deptno = 10
```

Objašnjenje

WHERE klauzula vam omogućava da preuzmete samo redove koji vas zanimaju. Ako je izraz u WHERE klauzuli tačan za bilo koji red, tada se taj red vraća.

Većina dobavljača podržava uobičajene operatore kao što su =, <, >, <=, >=, ! i <>. Pored toga, možda ćete želeti redove koji zadovoljavaju više uslova; to se može učiniti navođenjem AND, OR i zagradama, kao što je prikazano u sledećem receptu.

1.3 Pronalaženje redova koji zadovoljavaju više uslova

Zadatak

Želite da vratite redove koji zadovoljavaju više uslova.

Rešenje

Koristite klauzulu WHERE zajedno sa klauzulama OR i AND. Na primer, ako želite da pronađete sve zaposlene u odeljenju 10, zajedno sa zaposlenima koji zarade proviziju, zajedno sa zaposlenima u odeljenju 20 koji zarađuju do 2.000 USD:

```
1 select *
2   from emp
3  where deptno = 10
4     or comm is not null
5     or sal <= 2000 and deptno=20
```

Objašnjenje

Možete da koristite kombinaciju AND, OR i zagrade da biste vratili redove koji zadovoljavaju više uslova. U primeru rešenja, klauzula WHERE pronalazi redove takve da:

- DEPTNO je 10
- COMM nije NULL
- Plata je 2.000 USD ili manje za bilo kog zaposlenog u DEPTNO 20.

Prisustvo zagrada dovodi do zajedničkog vrednovanja uslova u njima.

Na primer, razmotrite kako se skup rezultata menja ako je upit napisan u zagradama kao što je ovde prikazano:

```
select *
 from emp
where (     deptno = 10
       or comm is not null
       or sal <= 2000
      )
  and deptno=20

EMPNO ENAME  JOB    MGR   HIREDATE     SAL        COMM  DEPTNO
----- ------ ----- -----  ----------- ----- ----------  ------
7369  SMITH  CLERK  7902  17-DEC-1980   800                 20
7876  ADAMS  CLERK  7788  12-JAN-1983  1100                 20
```

1.4 Preuzimanje podskupa kolona iz tabele

Zadatak

Imate tabelu i želite da vidite vrednosti za određene kolone, a ne za sve kolone.

Rešenje

Navedite kolone koje vas zanimaju. Na primer, da biste videli samo ime, broj odeljenja i platu zaposlenih:

```
1 select ename,deptno,sal
2   from emp
```

Objašnjenje

Zadavanjem kolona u klauzuli SELECT obezbeđujete da se ne vraćaju nikakvi suvišni podaci. Ovo može biti posebno važno pri preuzimanju podataka preko mreže, jer se izbegava gubljenje vremena svojstvenog preuzimanju podataka koji vam nisu potrebni.

1.5 Davanje smislenih imena za kolone

Zadatak

Želeli biste da promenite imena kolona koje se vraćaju vašim upitom tako da budu čitljivija i razumljivija. Razmotrite ovaj upit koji vraća zarade i provizije za svakog zaposlenog:

```
1 select sal,comm
2   from emp
```

Šta je SAL? Da li je skraćenica od *sale* (srp. *prodaja*)? Je li to nečije ime? Šta je COMM? Da li je to komunikacija? Želite da rezultati imaju značajnije oznake.

Rešenje

Da biste promenili imena rezultata upita, koristite AS ključnu reč u obliku *`original_name AS ovo_ime`*. Neke baze podataka ne zahtevaju AS, ali sve ga prihvataju:

```
 1 select sal as salary, comm as commission
 2   from emp

SALARY   COMMISSION
-------  ----------
    800
   1600         300
   1250         500
   2975
   1250        1400
   2850
   2450
   3000
   5000
   1500           0
   1100
    950
   3000
   1300
```

Objašnjenje

Korišćenje ključne reči AS za davanje novih imena kolonama koje vraća vaš upit poznato je kao davanje *aliasa (rezrevnog imena, pseudoimena)* kolonama. Stvaranje dobrih pseudonima može mnogo da pomogne da upit i njegovi rezultati budu razumljivi drugima.

1.6 Referenciranje kolone sa alijasom u klauzuli WHERE

Zadatak

Koristili ste pseudonime da biste obezbedili smislenija imena kolona za svoj rezultat i želeli biste da izuzmete neke redove pomoću klauzule WHERE. Međutim, vaš pokušaj pozivanja pseudoimena u klauzuli WHERE ne uspeva:

```
select sal as salary, comm as commission
from emp
where salary < 5000
```

Rešenje

Umotavanjem upita u umetnuti prikaz možete postaviti upit kolonama sa alijasima:

```
1 select *
2   from (
3 select sal as salary, comm as commission
4   from emp
5        ) x
6  where salary < 5000
```

Objašnjenje

U ovom jednostavnom primeru možete da izbegnete umetnuti prikaz i referencu COMM ili SAL direktno u klauzuli WHERE da biste postigli isti rezultat. Ovo rešenje vas upoznaje sa onim što biste trebali učiniti kada pokušavate da uputite bilo šta od sledećeg u klauzulu WHERE:

- Zbirne funkcije
- Skalarni podupiti
- Funkcije prozora
- Aliasi

Postavljanje vašeg upita, onoga koji daje pseudonime, u umetnuti prikaz daje vam mogućnost referenciranja na alias kolone u vašem spoljnom upitu. Zašto to treba da uradite? Klauzula WHERE procenjuje se pre SELECT; tako da SALARY i COMMISSION još uvek ne postoje kada se procenjuje WHERE klauzula upita „Zadatak". Ti pseudonimi se primenjuju tek nakon završetka obrade klauzule WHERE. Međutim, klauzula FROM procenjuje se pre WHERE. Postavljanjem originalnog upita u klauzulu FROM, rezultati tog upita generišu se ispred najudaljenije klauzule WHERE, a vaša krajnja klauzula WHERE „vidi" imena aliasa. Ova tehnika je posebno korisna kada stupci u tabeli nisu posebno dobro imenovani.



Umetnuti prikaz u ovom rešenju ima alias X. Ne zahtevaju sve baze podataka da umetnuti prikaz bude eksplicitno alias, ali neke to zahtevaju. Svi oni to prihvataju.

1.7 Nadovezivanje vrednosti kolona

Zadatak

Želite da vratite vrednosti iz više kolona kao jednu kolonu. Na primer, želeli biste da napravite ovaj skup rezultata iz upita prema EMP tabeli:

```
CLARK WORKS AS A MANAGER
KING WORKS AS A PRESIDENT
MILLER WORKS AS A CLERK
```

Međutim, podaci koji su vam potrebni za generisanje ovog skupa rezultata dolaze iz dve različite kolone, ENAME i JOB kolone u EMP tabeli:

```
 select ename, job
    from emp
    where deptno = 10

ENAME      JOB
---------- ---------
CLARK      MANAGER
KING       PRESIDENT
MILLER     CLERK</pre>
```

Rešenje

Pronađite i koristite ugrađenu funkciju koju pruža vaš DBMS za nadovezivanje vrednosti iz više kolona.

DB2, Oracle, PostgreSQL

Ove baze podataka koriste dvostruku vertikalnu traku kao operater nadovezivanja:

```
1 select ename||' WORKS AS A '||job as msg
2   from emp
3  where deptno=10
```

MiSQL

Ova baza podataka podržava funkciju CONCAT:

```
1 select concat(ename, ' WORKS AS A ',job) as msg
2   from emp
3  where deptno=10
```

SQL Server

Koristite + operator za nadovezivanje:

```
1 select ename + ' WORKS AS A ' + job as msg
2   from emp
3  where deptno=10
```

Objašnjenje

Koristite funkciju CONCAT za nadovezivanje vrednosti iz više kolona. Znak || je prečica za funkciju CONCAT u DB2, Oracle i PostgreSQL, dok je + prečica za SQL Server.

1.8 Korišćenje uslovne logike u SELECT izjavi

Zadatak

Želite da izvršite IF-ELSE operacije nad vrednostima u vašem SELECT izrazu. Na primer, želeli biste da napravite skup rezultata tako da ako je zaposleni plaćen 2.000 USD ili manje, vraća se poruka „UNDERPAID" (srp. *slabo plaćen*); ako je zaposlenom plaćeno 4.000 USD ili više, vraća se poruka „OVERPAID" (srp. *preplaćeno*); a ako su plaćeni negde između, vraća se „OK". Skup rezultata trebao bi izgledati ovako:

```
 ENAME            SAL  STATUS
---------- ----------  ---------
SMITH             800  UNDERPAID
ALLEN            1600  UNDERPAID
WARD             1250  UNDERPAID
JONES            2975  OK
MARTIN           1250  UNDERPAID
BLAKE            2850  OK
CLARK            2450  OK
SCOTT            3000  OK
KING             5000  OVERPAID
TURNER           1500  UNDERPAID
ADAMS            1100  UNDERPAID
JAMES             950  UNDERPAID
FORD             3000  OK
MILLER           1300  UNDERPAID</pre>
```

Rešenje

Koristite CASE izraz za izvođenje uslovne logike direktno u vašem SELECT izrazu:

```
1 select ename,sal,
2        case when sal <= 2000 then 'UNDERPAID'
3             when sal >= 4000 then 'OVERPAID'
4             else 'OK'
5        end as status
6   from emp
```

Objašnjenje

CASE izraz vam omogućava da izvodite logiku uslova na vrednostima koje vraća upit. Možete dodati pseudonim za izraz CASE da biste vratili čitljiviji skup rezultata. U rešenju ćete videti pseudonim STATUS dat rezultatu izraza CASE. Klauzula ELSE nije obavezna. Izostavite ELSE i izraz CASE će vratiti NULL za bilo koji red koji ne zadovoljava uslov testa.

1.9 Ograničavanje broja vraćenih redova

Zadatak

Želite da ograničite broj redova vraćenih vašim upitom. Ne zanima vas redosled; bilo kojih *n* redova vam je potrebo.

Rešenje

Koristite ugrađenu funkciju vaše baze podataka za kontrolu broja vraćenih redova.

DB2

U DB2 koristite klauzulu FETCH FIRST:

```
1 select *
2   from emp fetch first 5 rows only
```

MiSQL i PostgreSQL

Uradite isto u MiSQL-u i PostgreSQL-u koristeći LIMIT:

```
1 select *
2   from emp limit 5
```

Oracle

U Oracleu postavite ograničenje na broj vraćenih redova upotrebom ROWNUM u klauzuli WHERE:

```
1 select *
2   from emp
3  where rownum <= 5
```

SQL Server

Koristite rezervisanu reč TOP da biste ograničili broj vraćenih redova:

```
1 select top 5 *
2   from emp
```

Objašnjenje

Mnogi dobavljači pružaju klauzule kao što su FETCH FIRST i LIMIT koje vam omogućavaju da zadate broj redova koji se vraćaju upitom. Oracle je drugačiji, jer morate koristiti funkciju ROWNUM koja vraća broj za svaki vraćeni red (rastuće vrednosti počev od 1).

Evo šta se dešava kada koristite ROWNUM <= 5 za vraćanje prvih pet redova:

1. Oracle izvršava vaš upit.
2. Oracle preuzima prvi red i naziva ga red broj 1.

3. Da li smo prešli red broj 5? Ako nismo, tada Oracle vraća red jer ispunjava kriterijume da bude numerisan manje ili jednako 5. Ako je odgovor da, Oracle ne vraća red.
4. Oracle preuzima sledeći red i povećava broj reda (na 2, zatim na 3, pa na 4 i tako dalje).
5. Ide na korak 3.

Kao što ovaj proces pokazuje, vrednosti Oracleovog ROWNUM dodeljuju se *posle* preuzimanja svakog reda. Ovo je važna i ključna tačka. Mnogi Oracle programeri pokušavaju da vrate samo, recimo, peti red navodeći ROWNUM = 5.

Korišćenje uslova jednakosti u sprezi sa ROWNUM-om je loša ideja. Evo šta se dešava kada pokušate da vratite, recimo, peti red koristeći ROWNUM = 5:

1. Oracle izvršava vaš upit.
2. Oracle preuzima prvi red i naziva ga red broj jedan.
3. Jesmo li već stigli do petog reda? Ako nismo, tada Oracle odbacuje red jer ne ispunjava kriterijume. Ako je odgovor da, Oracle vraća red. Ali odgovor nikada neće biti da!
4. Oracle preuzima sledeći red i naziva ga red broj jedan. To je zato što prvi red koji se vraća iz upita mora biti numerisan sa 1.
5. Ide na korak 3.

Pažljivo proučite ovaj proces i videćete zašto upotreba ROWNUM = 5 za vraćanje petog reda ne uspeva. Ne možete imati peti red ako prvo ne vratite redove jedan do četiri!

Možda ćete primetiti da ROWNUM = 1 u stvari radi na vraćanju prvog reda, što može izgledati u suprotnosti sa dosadašnjim objašnjenjem. Razlog zašto ROWNUM = 1 vraćan prvi red je taj što Oracle mora da pokuša da preuzme barem jednom da bi utvrdio da li postoje redovi u tabeli. Pažljivo pročitajte prethodni postupak, zamenjujući 1 za 5, i razumećete zašto je u redu zadati ROWNUM = 1 kao uslov (za vraćanje jednog reda).

1.10 Vraćanje n nasumičnih zapisa iz tabele

Zadatak

Iz tabele želite da vratite određeni broj nasumičnih zapisa. Sledeću izjavu želite da izmenite tako da će uzastopna izvršavanja proizvesti drugačiji skup od pet redova:

```
select ename, job
from emp
```

Rešenje

Uzmite bilo koju ugrađenu funkciju koju vaš DBMS podržava za vraćanje nasumičnih vrednosti. Koristite tu funkciju u klauzuli ORDER BY da biste nasumično sortirali redove. Zatim, koristite tehniku prethodnog recepta da biste ograničili broj nasumično sortiranih redova koji se vraćaju.

DB2

Koristite ugrađenu funkciju RAND u saradnji sa ORDER BY i FETCH:

```
1 select ename,job
2   from emp
3  order by rand() fetch first 5 rows only
```

MySQL

Koristite ugrađenu funkciju RAND u kombinaciji sa LIMIT i ORDER BY:

```
1 select ename,job
2   from emp
3  order by rand() limit 5
```

PostgreSQL

Koristite ugrađenu funkciju RANDOM u kombinaciji sa LIMIT i ORDER BY:

```
1 select ename,job
2   from emp
3  order by random() limit 5</pre>
```

Oracle

Koristite ugrađenu funkciju VALUE, koja se nalazi u ugrađenom paketu DBMS_RANDOM, zajedno sa ORDER BY i ugrađenom funkcijom ROWNUM:

```
1 select *
2   from (
3  select ename, job
4    from emp
6   order by dbms_random.value()
7        )
8   where rownum <= 5
```

SQL Server

Koristite ugrađenu funkciju NEWID zajedno sa TOP i ORDER BY da biste vratili slučajni skup rezultata:

```
1 select top 5 ename,job
2   from emp
3  order by newid()
```

Objašnjenje

Klauzula ORDER BY može prihvatiti povratnu vrednost funkcije i koristiti je za promenu redosleda skupa rezultata. Sva ova rešenja ograničavaju broj redova za vraćanje *nakon* što se izvrši funkcija u klauzuli ORDER BY. Korisnicima koji nisu Oracle može biti korisno pogledati Oracle rešenje jer ono pokazuje (konceptualno) šta se dešava ispod haube ostalih rešenja.

Važno je da ne pomešate upotrebu funkcije u klauzuli ORDER BY sa upotrebom numeričke konstante. Kada zadate numeričku konstantu u klauzuli ORDER BY, zahtevate da se sortiranje izvrši prema koloni u na tom rednom položaju na listi SELECT. Kada zadate funkciju u klauzuli ORDER BY, sortiranje se vrši na rezultatu iz funkcije, kako se ona izračunava za svaki red.

1.11 Pronalaženje null vrednosti

Zadatak

Želite da pronađete sve redove koji su null u određenoj koloni.

Rešenje

Da biste utvrdili da li je vrednost null, morate koristiti IS NULL:

```
1 select *
2   from emp
3  where comm is null
```

Objašnjenje

NULL nikada nije jednak a nije ni nejednak nečemu, čak ni sebi; stoga ne možete koristiti = ili != za testiranje da li je u koloni NULL. Da biste utvrdili da li red ima NULL vrednosti, morate koristiti IS NULL. Takođe možete da koristite IS NOT NULL da biste pronašli redove bez null u datoj koloni.

1.12 Pretvaranje null u stvarne vrednosti

Zadatak

Imate redove koji sadrže null i želeli bi da vrate nenull vrednosti umesto tih null.

Rešenja

Koristite funkciju COALESCE da biste null zamenili realnim vrednostima:

```
1 select coalesce(comm,0)
2   from emp
```

Objašnjenje

Funkcija COALESCE uzima jednu ili više vrednosti kao argumente. Funkcija vraća prvu nenull vrednost sa liste. U rešenju vrednost COMM se vraća kad god COMM nije null. U suprotnom, vraća se nula (0).

Kada radite sa null, najbolje je iskoristiti ugrađenu funkcionalnost koju pruža vaš DBMS; u mnogim slučajevima naći ćete da nekoliko funkcija jednako dobro radi i ovaj zadatak. COALESCE funkcioniše za sve DBMS-ove. Pored toga, CASE se može koristiti i za sve DBMS-ove:

```
select case
       when comm is not null then comm
       else 0
       end
  from emp
```

Iako možete da koristite CASE za prevođenje nullova u vrednosti, možete videti da je mnogo lakše i sažetije koristiti COALESCE.

1.13 Traženje obrazaca

Zadatak

Želite da vratite redove koji se podudaraju sa određenim podnizom ili šablonom. Uzmite u obzir sledeći upit i skup rezultata:

```
select ename, job
  from emp
 where deptno in (10,20)

ENAME      JOB
---------- ---------
SMITH      CLERK
JONES      MANAGER
CLARK      MANAGER
SCOTT      ANALYST
KING       PRESIDENT
ADAMS      CLERK
FORD       ANALYST
MILLER     CLERK
```

Od zaposlenih u odeljenjima 10 i 20 želite da vratite samo one koji negde u svom imenu imaju ili „I” ili naziv posla koji se završava sa „ER”:

```
ENAME      JOB
---------- ---------
SMITH      CLERK
JONES      MANAGER
CLARK      MANAGER
KING       PRESIDENT
MILLER     CLERK
```

Rešenje

Koristite operator LIKE zajedno sa SQL džoker operatorom (%):

```
1 select ename, job
2   from emp
3  where deptno in (10,20)
4    and (ename like '%I%' or job like '%ER')
```

Objašnjenje

Kada se LIKE koristi u operaciji podudaranja šablona, operator procenat (%) se podudara sa bilo kojim nizom znakova. Većina SQL implementacija takođe obezbeđuje donju crtu ("_") za podudaranje sa jednim znakom. Zatvaranjem šablona za pretragu „I" sa operatorom %, biće vraćen bilo koji niz koji sadrži „I" (na bilo kojoj poziciji). Ako šablon pretraživanja ne zatvorite sa %, tada će mesto na kojem postavite operator uticati na rezultate upita. Na primer, da biste pronašli naslove poslova koji se završavaju sa „ER", dodajte operatoru % ispred „ER"; ako je zahtev da se pretražuju svi poslova koji počinju sa „ER", dodajte operator % iza „ER".

1.14 Rezime

Ovi recepti su možda jednostavni, ali su i osnovni. Nalaženje informacija je srž upita za baze podataka, a to znači da su ovi recepti u osnovi gotovo svega o čemu se raspravlja u knjizi.