KRATAK SADRŽAJ

Zašto je korisno učiti SQL? 1
Uloga SQL developera 8

DEO I

UVOD 21

POGLAVLJE 1
CREATE TABLE naredba 25

POGLAVLJE 2
ALTER TABLE naredba 31

POGLAVLJE 3
DROP TABLE naredba 35

POGLAVLJE 4
TRUNCATE TABLE naredba 38

POGLAVLJE 5
Komentari 41

POGLAVLJE 6
RENAME TABLE naredba 43

POGLAVLJE 7
SELECT naredba. 47

POGLAVLJE 8
INSERT INTO naredba. 50

POGLAVLJE 9

SELECT INTO naredba 53

POGLAVLJE 10
INSERT INTO SELECT naredba 56

POGLAVLJE 11
SELECT DISTINCT naredba. 59

POGLAVLJE 12
DELETE naredba. 62

POGLAVLJE 13
SELECT TOP klauzula 65

POGLAVLJE 14
WHERE klauzula 68

POGLAVLJE 15
ORDER BY ključna reč 71

POGLAVLJE 16
GROUP BY klauzula 74

POGLAVLJE 17
TRUNCATE TABLE naredba 77

POGLAVLJE 18
RENAME TABLE naredba 80

POGLAVLJE 19
UPDATE naredba 84

POGLAVLJE 20
COUNT() funkcija 87

POGLAVLJE 21
UPDATE naredba 90

POGLAVLJE 22
SQL operatori 93

POGLAVLJE 23
ANY i ALL operatori. 98

POGLAVLJE 24

AND operator 101

POGLAVLJE 25
BETWEEN operator 105

POGLAVLJE 26
EXISTS operator 108

POGLAVLJE 27
In operator 111

POGLAVLJE 28
LIKE operator 114

POGLAVLJE 29
NOT operator 118

POGLAVLJE 30
OR operator 121

POGLAVLJE 31
SOME operator 124

POGLAVLJE 32
UNION operator 127

POGLAVLJE 33
AVG() funkcija 130

POGLAVLJE 34
NULL funkcije 133

POGLAVLJE 35
SUM() funkcija 137

POGLAVLJE 36
MIN() i MAX() funkcije 140

POGLAVLJE 37
CASE izraz 143

POGLAVLJE 38
HAVING klauzula 147

POGLAVLJE 39
Uvod u ograničenja vezana za podatke u tabeli 150

POGLAVLJE 40
CHECK ograničenje 153

POGLAVLJE 41
DEFAULT ograničenje 157

POGLAVLJE 42
FOREIGN KEY ograničenje. 160

POGLAVLJE 43
NOT NULL ograničenje 164

POGLAVLJE 44
PRIMARY KEY ograničenje 167

POGLAVLJE 45
UNIQUE ograničenje 171

POGLAVLJE 46
NULL vrednosti 174

POGLAVLJE 47
AUTO INCREMENT polje. 177

POGLAVLJE 48
Rad sa datumima 180

POGLAVLJE 49
POGLEDI 184

POGLAVLJE 50
Zapamćene procedure 191

POGLAVLJE 51
Umetanje SQL koda 195

POGLAVLJE 52
Uvod u baze podataka 199

POGLAVLJE 53
CREATE DATABASE naredba 205

POGLAVLJE 54
BACKUP DATABASE naredba 208

POGLAVLJE 55
DROP DATABASE komanda 211

POGLAVLJE 56
CREATE INDEX naredba 214

POGLAVLJE 57
GRANT naredba. 217

POGLAVLJE 58
REVOKE naredba 219

POGLAVLJE 59
Pseudonimi. 222

POGLAVLJE 60
Wildcard karakteri 225

POGLAVLJE 61
JOIN klauzula. 228

POGLAVLJE 62
LEFT JOIN klauzula 232

POGLAVLJE 63
FULL OUTER JOIN klauzula 235

POGLAVLJE 64
RIGHT JOIN klauzula 238

POGLAVLJE 65
INNER JOIN klauzula 241

POGLAVLJE 66
SELF JOIN klauzula 244

POGLAVLJE 67
SQL ključne reči 247

POGLAVLJE 68
Tipovi podataka u SQL programskom jeziku 251

DEO II

POGLAVLJE 69
Uvod 263

POGLAVLJE 70
SQLgpt i vežbe sa upitima 285

POGLAVLJE 71
Pitanja za intervju za posao 358

POGLAVLJE 72
Pomoćne tabele (CTE) u SQL programskom jeziku. 372

POGLAVLJE 73
SQLgpt odgovori na pitanja kojih nema u knjizi 396

SADRŽAJ

Zašto je korisno učiti SQL? 1

Univerzalni jezik za upravljanje bazama podataka ... 2
Važnost u analizi podataka ... 2
Fleksibilnost i svestranost ... 3
Tražena veština na tržištu rada ... 3
Osnova za naprednije tehnologije ... 4
Podrška za donošenje odluka i izveštavanje ... 5
Efikasnost u radu s podacima ... 5
Budućnost SQL jezika ... 6

Uloga SQL developera 8

SQL developer kao analitičar podataka ... 9
Uloga SQL developera u razvoju softvera ... 10
Uloga SQL developera u razvoju baza podataka ... 11
Uloga SQL developera u manipulacijama podacima ... 12
Saradnja SQL developera sa naučnicima ... 13
Uloga SQL developera na zaštiti podataka ... 13
SQL developeri i upravljanje horizontalnim particioniranje i repliciranjem ... 14
SQL develoreri i računarstvo u oblaku ... 15
SQL developeri i agilni razvoj softvera ... 16
SQL developeri i integracija podataka sa drugim programskim jezicima ... 17
SQL developeri i integritet podataka ... 18

DEO I

UVOD 21

DDL naredbe ... 22
DML naredbe ... 22
DCL naredbe ... 23
TCL naredbe ... 23
DQL naredba ... 24

POGLAVLJE 1

CREATE TABLE naredba 25

Uvod i sintaksa....25
Kako funkcioniše....26
Primeri upotrebe....26
Performanse....27
Ograničenja i zablude....27
Zaključak....28
Kviz....28
Kreiranje nove tabele na osnovu postojeće....29
Zaključak....30

POGLAVLJE 2

ALTER TABLE naredba 31

Uvod i sintaksa....31
Kako funkcioniše....32
Primeri upotrebe....32
Performanse....32
Ograničenja i zablude....33
Zaključak....33
Kviz....34

POGLAVLJE 3

DROP TABLE naredba 35

Uvod i sintaksa....35
Kako funkcioniše....35
Primeri upotrebe....35
Performanse....36
Ograničenja i zablude....36
Zaključak....37
Kviz....37

POGLAVLJE 4

TRUNCATE TABLE naredba 38

Uvod i sintaksa....38
Kako funkcioniše....38
Primeri upotrebe....38
Performanse....40
Ograničenja i zablude....40
Zaključak....40

POGLAVLJE 5

Komentari 41

Saveti i najbolja praksa....41
Komentari u realnim scenarijima....42

POGLAVLJE 6

RENAME TABLE naredba 43

Uvod i sintaksa....43
Kako funkcioniše....43
Primeri upotrebe....44
Ograničenja i zablude....45
Zaključak....46

POGLAVLJE 7

SELECT naredba. 47

Uvod i sintaksa....47
Kako funkcioniše....47
Primeri upotrebe....47
Performanse....48
Ograničenja i zablude....48
Zaključak....49
Kviz....49

POGLAVLJE 8

INSERT INTO naredba. 50

Uvod i sintaksa....50
Kako funkcioniše....50
Primeri upotrebe....50
Performanse....51
Ograničenja i zablude....51
Zaključak....52
Kviz....52

POGLAVLJE 9

SELECT INTO naredba 53

Uvod i sintaksa....53
Kako funkcioniše....53
Primeri upotrebe....54
Performanse....54
Ograničenja i zablude....55
Zaključak....55
Kviz....55

POGLAVLJE 10

INSERT INTO SELECT naredba 56

Uvod i sintaksa....56
Kako funkcioniše....56
Primeri upotrebe....56
Performanse....57
Ograničenja i zablude....57
Zaključak....58
Kviz....58

POGLAVLJE 11

SELECT DISTINCT naredba. 59

Uvod i sintaksa....59
Kako funkcioniše....59
Primeri upotrebe....59
Performanse....60
Ograničenja i zablude....60
Zaključak....61
Kviz....61

POGLAVLJE 12

DELETE naredba. 62

Uvod i sintaksa....62
Kako funkcioniše....62
Primeri upotrebe....62
Performanse....63
Ograničenja i zablude....63
Zaključak....64
Kviz....64

POGLAVLJE 13

SELECT TOP klauzula 65

Uvod i sintaksa....65
Kako funkcioniše....65
Primeri upotrebe....65
Ograničenja i zablude....66
Zaključak....67
Kviz....67

POGLAVLJE 14

WHERE klauzula 68

Uvod i sintaksa....68
Kako funkcioniše....68
Primeri upotrebe....68
Performanse....69
Ograničenja i zablude....69
Zaključak....70
Kviz....70

POGLAVLJE 15

ORDER BY ključna reč 71

Uvod i sintaksa....71
Kako funkcioniše....71
Primeri upotrebe....72
Performanse....72
Ograničenja i zablude....72

Zaključak....73
Kviz....73

POGLAVLJE 16

GROUP BY klauzula 74

Uvod i sintaksa....74
Kako funkcioniše....74
Primeri upotrebe....74
Performanse....75
Ograničenja i zablude....75
Zaključak....76
Kviz....76

POGLAVLJE 17

TRUNCATE TABLE naredba 77

Uvod i sintaksa....77
Kako funkcioniše....77
Primeri upotrebe....77
Performanse....79
Ograničenja i zablude....79
Zaključak....79

POGLAVLJE 18

RENAME TABLE naredba 80

Uvod i sintaksa....80
Kako funkcioniše....80
Primeri upotrebe....81
Ograničenja i zablude....82
Zaključak....83

POGLAVLJE 19

UPDATE naredba 84

Uvod i sintaksa....84
Kako funkcioniše....84
Primeri upotrebe....84
Performanse....85
Ograničenja i zablude....85
Zaključak....86
Kviz....86

POGLAVLJE 20

COUNT() funkcija 87

Uvod i sintaksa....87
Kako funkcioniše....87
Primeri upotrebe....87
Performanse....88

Ograničenja i zablude 88
Zaključak 89
Kviz 89

POGLAVLJE 21

UPDATE naredba 90

Uvod i sintaksa 90
Kako funkcioniše 90
Primeri upotrebe 90
Performanse 91
Ograničenja i zablude 91
Zaključak 92
Kviz 92

POGLAVLJE 22

SQL operatori 93

Aritmetički operatori 93
Operatori nad bitovima 94
SQL operatori za poređenje 95
SQL složeni operatori 96
SQL logički operatori 97

POGLAVLJE 23

ANY i ALL operatori. 98

Uvod i sintaksa 98
Kako funkcionišu 98
Primeri upotrebe 99
Performanse 99
Ograničenja i zablude 99
Zaključak 100
Kviz 100

POGLAVLJE 24

AND operator 101

Uvod i sintaksa 101
Kako funkcioniše 101
Primeri upotrebe 101
Performanse 102
Ograničenja i zablude 102
Zaključak 103
Kviz 103

POGLAVLJE 25

BETWEEN operator 105

Uvod i sintaksa 105
Kako funkcioniše 105
Primeri upotrebe 105

Performanse....106
Ograničenja i zablude....106
Zaključak....107
Kviz....107

POGLAVLJE 26

EXISTS operator 108

Uvod i sintaksa....108
Kako funkcioniše....108
Primeri upotrebe....108
Performanse....109
Ograničenja i zablude....109
Zaključak....110
Kviz....110

POGLAVLJE 27

In operator. 111

Uvod i sintaksa....111
Kako funkcioniše....111
Primeri upotrebe....111
Performanse....112
Ograničenja i zablude....112
Zaključak....113
Kviz....113

POGLAVLJE 28

LIKE operator. 114

Uvod i sintaksa....114
Kako funkcioniše....114
Primeri upotrebe....115
Performanse....115
Ograničenja i zablude....116
Kviz....117

POGLAVLJE 29

NOT operator. 118

Uvod i sintaksa....118
Kako funkcioniše....118
Primeri upotrebe....118
Performanse....119
Ograničenja i zablude....119
Zaključak....120
Kviz....120

POGLAVLJE 30

OR operator 121

Uvod i sintaksa....121
Kako funkcioniše....121
Primeri upotrebe....121
Performanse....122
Ograničenja i zablude....122
Zaključak....123
Kviz....123

POGLAVLJE 31

SOME operator 124

Uvod i sintaksa....124
Kako funkcioniše....124
Primeri upotrebe....124
Performanse....125
Ograničenja i zablude....125
Zaključak....126
Kviz....126

POGLAVLJE 32

UNION operator 127

Uvod i sintaksa....127
Kako funkcioniše....127
Primeri upotrebe....127
Performanse....128
Ograničenja i zablude....128
Zaključak....129
Kviz....129

POGLAVLJE 33

AVG() funkcija 130

Uvod i sintaksa....130
Kako funkcioniše....130
Primeri upotrebe....130
Performanse....131
Ograničenja i zablude....132
Zaključak....132
Kviz....132

POGLAVLJE 34

NULL funkcije. 133

Uvod i sintaksa....133
Kako funkcionišu....133
Primeri upotrebe....133
Performanse....134

Ograničenja i zablude 135
Zaključak 136
Kviz 136

POGLAVLJE 35

SUM() funkcija 137

Uvod i sintaksa 137
Kako funkcioniše 137
Primeri upotrebe 138
Performanse 138
Ograničenja i zablude 138
Zaključak 139
Kviz 139

POGLAVLJE 36

MIN() i MAX() funkcije. 140

Uvod i sintaksa 140
Kako funkcioniše 140
Primeri upotrebe 140
Performanse 141
Ograničenja i zablude 141
Zaključak 142
Kviz 142

POGLAVLJE 37

CASE izraz 143

Uvod i sintaksa 143
Kako funkcioniše 143
Primeri upotrebe 144
Performanse 144
Zaključak 145
Kviz 146

POGLAVLJE 38

HAVING klauzula 147

Uvod i sintaksa 147
Kako funkcioniše 147
Primeri upotrebe 148
Performanse 148
Ograničenja i zablude 149
Zaključak 149
Kviz 149

POGLAVLJE 39

Uvod u ograničenja vezana za podatke u tabeli 150

Uvod i sintaksa 150

Kako funkcionišu....151
Primeri upotrebe....151
Performanse....151
Ograničenja i zablude....152
Zaključak....152
Kviz....152

POGLAVLJE 40

CHECK ograničenje 153

Uvod i sintaksa....153
Kako funkcioniše....153
Primeri upotrebe....154
Performanse....154
Ograničenja i zablude....154
Zaključak....155
Kviz....155
DROP CHECK ograničenja....155
Uvod i sintaksa....155
Kako funkcioniše....156

POGLAVLJE 41

DEFAULT ograničenje 157

Uvod i sintaksa....157
Kako funkcioniše....157
Primeri upotrebe....158
Performanse....158
Ograničenja i zablude....158
Zaključak....159
Kviz....159

POGLAVLJE 42

FOREIGN KEY ograničenje. 160

Uvod i sintaksa....160
Kako funkcioniše....160
Primeri upotrebe....161
Performanse....161
Ograničenja i zablude....162
Zaključak....162
Kviz....162
DROP FOREIGN KEY ograničenje....163
Uvod i sintaksa....163
Kako funkcioniše....163

POGLAVLJE 43

NOT NULL ograničenje 164

Uvod i sintaksa....164
Kako funkcioniše....164

Primeri upotrebe ... 165
Performanse ... 165
Ograničenja i zablude ... 165
Zaključak ... 166
Kviz ... 166

POGLAVLJE 44

PRIMARY KEY ograničenje 167

Uvod i sintaksa ... 167
Kako funkcioniše ... 167
Primeri upotrebe ... 168
Performanse ... 168
Ograničenja i zablude ... 168
Zaključak ... 169
Kviz ... 169
Uklanjanje PRIMARY KEY ograničenja ... 169
Uvod i sintaksa ... 169
Kako funkcioniše ... 170
Ograničenja i zablude ... 170

POGLAVLJE 45

UNIQUE ograničenje 171

Uvod i sintaksa ... 171
Kako funkcioniše ... 171
Primeri upotrebe ... 172
Performanse ... 172
Ograničenja i zablude ... 172
Zaključak ... 173
Kviz ... 173

POGLAVLJE 46

NULL vrednosti 174

Uvod i sintaksa ... 174
Kako funkcioniše ... 174
Primeri upotrebe ... 175
Performanse ... 175
Ograničenja i zablude ... 175
Zaključak ... 176
Kviz ... 176

POGLAVLJE 47

AUTO INCREMENT polje. 177

Uvod i sintaksa ... 177
Kako funkcioniše ... 177
Primeri upotrebe ... 178
Performanse ... 178

Ograničenja i zablude 178
Zaključak 179
Kviz 179

POGLAVLJE 48

Rad sa datumima 180

Uvod i sintaksa 180
Kako funkcioniše 180
Primeri upotrebe 180
Performanse 181
Ograničenja i zablude 181
Zaključak 182
Kviz 182
Upoređivanje dva datuma bez vremenske komponente 182
Kako funkcioniše 182
Primer 182
Primer upita 182
Objašnjenje 183
Značaj upoređivanja datuma 183
Dodatni saveti 183

POGLAVLJE 49

POGLEDI 184

Uvod i sintaksa 184
Kako funkcioniše 184
Primeri upotrebe 184
Performanse 185
Ograničenja i zablude 185
Zaključak 186
Modifikacija pogleda 186
Uvod i sintaksa 186
Kako funkcioniše 186
Primeri upotrebe 186
Zaključak 187
CREATE OR REPLACE VIEW 187
Uvod i sintaksa 187
Kako funkcioniše 188
Primeri upotrebe 188
Zaključak 189
Brisanje pogleda 189
Uvod i sintaksa 189
Kako funkcioniše 189
Posledice brisanja pogleda 189
Zaključak 190

POGLAVLJE 50

Zapamćene procedure 191

Uvod i sintaksa....191
Kako funkcioniše....191
Primeri upotrebe....192
Performanse....192
Ograničenja i zablude....193
Zaključak....194
Kviz....194

POGLAVLJE 51

Umetanje SQL koda 195

Šta je umetanje SQL koda i zašto je bitno....195
Kako dolazi do napada umetanjem SQL koda....195
Primeri napada umetanjem SQL koda i njihove posledice....195
Najbolje prakse za prevenciju umetanja SQL koda....196
Preporuke za sigurne prakse kodiranja....196
Alati i tehnike za otkrivanje i ublažavanje napada umetanjem SQL koda....196
Resursi za dalje učenje o umetanju SQL koda....196
Tipovi napada umetanjem SQL koda....197
Prevencija i zaštita....198

POGLAVLJE 52

Uvod u baze podataka 199

Koji tipovi baza podataka postoje....200
Koji sistemi za rad sa bazama podataka se najviše koriste u svetu?....201
Kako se SQL i njegovo učenje uklapa u svaki od ovih sistema?....202
Koliko se svaki od ovih sistema razlikuje od standardnog SQL jezika?....203
Da li je dobro da prvo naučim SQL jezik, a tek onda neki od sistema ili da odmah pređem na učenje nekog od njih?....204

POGLAVLJE 53

CREATE DATABASE naredba 205

Uvod i sintaksa....205
Kako funkcioniše....205
Primeri upotrebe....205
Performanse....206
Ograničenja i zablude....206
Zaključak....207
Kviz....207

POGLAVLJE 54

BACKUP DATABASE naredba 208

Uvod i sintaksa....208
Kako funkcioniše....208
Primeri upotrebe....209

Performanse....209
Ograničenja i zablude....209
Zaključak....210
Kviz....210

POGLAVLJE 55

DROP DATABASE komanda....211

Uvod i sintaksa....211
Kako funkcioniše....211
Primeri upotrebe....211
Performanse....212
Ograničenja i zablude....212
Zaključak....213
Kviz....213

POGLAVLJE 56

CREATE INDEX naredba....214

Uvod i sintaksa....214
Kako funkcioniše....214
Primeri upotrebe....214
Performanse....215
Ograničenja i zablude....215
Zaključak....216
Kviz....216
DROP INDEX naredba....216
Kako funkcioniše....216

POGLAVLJE 57

GRANT naredba....217

Uvod i sintaksa....217
Kako funkcioniše....217
Primeri upotrebe....217
Performanse....218
Ograničenja i zablude....218
Zaključak....218
Kviz....218

POGLAVLJE 58

REVOKE naredba....219

Uvod i sintaksa....219
Kako funkcioniše....219
Primeri upotrebe....219
Performanse....220
Ograničenja i zablude....220
Zaključak....221
Kviz....221

POGLAVLJE 59

Pseudonimi 222

Uvod i sintaksa....222
Kako funkcioniše....222
Primeri upotrebe....222
Performanse....223
Ograničenja i zablude....223
Zaključak....224
Kviz....224

POGLAVLJE 60

Wildcard karakteri 225

Uvod i sintaksa....225
Kako funkcioniše....225
Primeri upotrebe....225
Performanse....226
Ograničenja i zablude....227
Zaključak....227
Kviz....227

POGLAVLJE 61

JOIN klauzula. 228

Uvod i sintaksa....228
Kako funkcioniše....228
Primeri upotrebe....229
Performanse....229
Ograničenja i zablude....230
Zaključak....231
Kviz....231

POGLAVLJE 62

LEFT JOIN klauzula 232

Uvod i sintaksa....232
Kako funkcioniše....232
Primeri upotrebe....232
Performanse....233
Ograničenja i zablude....233
Zaključak....234
Kviz....234

POGLAVLJE 63

FULL OUTER JOIN klauzula 235

Uvod i sintaksa....235
Kako funkcioniše....235
Primeri upotrebe....235
Performanse....236

Ograničenja i zablude 236
Zaključak 237
Kviz 237

POGLAVLJE 64

RIGHT JOIN klauzula 238

Uvod i sintaksa 238
Kako funkcioniše 238
Performanse 239
Ograničenja i zablude 239
Zaključak 240
Kviz 240

POGLAVLJE 65

INNER JOIN klauzula 241

Uvod i sintaksa 241
Kako funkcioniše 241
Primeri upotrebe 241
Performanse 242
Ograničenja i zablude 242
Zaključak 243
Kviz 243

POGLAVLJE 66

SELF JOIN klauzula 244

Uvod i sintaksa 244
Kako funkcioniše 244
Primeri upotrebe 244
Performanse 245
Zaključak 246
Kviz 246

POGLAVLJE 67

SQL ključne reči 247

POGLAVLJE 68

Tipovi podataka u SQL programskom jeziku 251

Primeri upotrebe 253
CHAR(n) 253
VARCHAR(50) 253
TEXT 253
INTEGER 254
SMALLINT 254
BIGINT 254
DECIMAL(p, s) / NUMERIC(p, s) 255
FLOAT(n) 255

REAL ... 256
DATE ... 256
TIME ... 256
DATETIME ... 257
TIMESTAMP ... 257
BOOLEAN ... 258
BINARY(n) ... 258
VARBINARY(n) ... 258
BLOB ... 259
ENUM ... 259
SET ... 260

DEO II

POGLAVLJE 69

Uvod 263

POGLAVLJE 70

SQLgpt i vežbe sa upitima 285

Uvod ... 285
Zadatak broj 1 – Svi zaposleni koji su radili za više od 3 klijenta ... 285
Zadatak broj 2 – Svi studenti koji su dobili ocene veće od 8 ... 287
Zadatak broj 3 – Zaposleni koji u odeljenju rade duže od 5 godina ... 288
Zadatak broj 4 – Proizvodi koji nisu naručeni ... 289
Zadatak broj 5 – Klijenti koji su potrošili više od 10.000 dinara ... 290
Zadatak broj 6 -Pacijenti sa više od 3 posete ustanovi ... 291
Zadatak broj 7 – Proizvodi čije recenzije su manje od 3 ... 292
Zadatak broj 8 – Zaposleni koji su radili za više od 3 klijenta ... 293
Zadatak broj 9 – Artikli kojima je porasla prodaja u poslednje dve godine ... 294
Zadatak broj 10 – Korisnici koji su pazarili više od prosečnih kupovina ... 295
Zadatak broj 11 – Studenti koji nisu položili ispit ... 296
Zadatak broj 12- Zaposleni koji nisu radili na projektima u inostranstvu ... 297
Zadatak broj 13 – Kreiranje arhive prodaje ... 298
Zadatak broj 14 – Ažuriranje tabele dodavanjem novih zapisa ... 299
Zadatak broj 15 – Bonus bodovi svim studentima sa prosečnim ocenama većim od 9 ... 300
Zadatak broj 16 – Zaposleni koji nemaju dodeljen projekat ... 301
Zadatak broj 17 – Studenti koji nisu podneli nijednu prijavu za stipendiju ... 301
Zadatak broj 18 – Rang lista studenata ... 302
Zadatak broj 19 – Projekti koji premašuju budžet ... 303
Zadatak broj 20 – Ažuriranje statusa narudžbina ... 305
Zadatak broj 21 – Ažuriranje statusa zaposlenih na određeni nivo ... 306
Zadatak broj 22 – Ažuriranje kontaktnih informacija klijenata ... 307
Zadatak broj 23 – Implementacija logike za automatsko promovisanje zaposlenih ... 308
Zadatak broj 24 – Optimizovanje zaliha ažuriranjem minimalnih količina ... 309
Zadatak broj 25 – Izveštaj o zaposlenima i svim klijentima ... 310
Zadatak broj 26 – Spisak zaposlenih i klijenata kojima je rođendan određenog meseca ... 311
Zadatak broj 27 – Dizajniranje strukture tabele Skladište ... 312

Zadatak broj 28 – Ažuriranje strukture tabele Zaposleni 313
Zadatak broj 29 – Rangiranje na osnovu određenog kriterijuma 314
Zadatak broj 30 – Identifikovanje najboljih 10% studenata 315
Zadatak broj 31 – Spisak 5 najskupljih proizvoda unutar kategorije 316
Zadatak broj 32 – Prvih 5 zaposlenih sa najdužim stažom 317
Zadatak broj 33 – Proizvod sa drugom najvišom cenom 318
Zadatak broj 34 – Selekcija prvih 50% proizvoda unutar kategorije 319
Zadatak broj 35 – Poslednih 25% narudžbina po danu 321
Zadatak 36 – Analiza ukupne prodaje po godini, kvartalu i mesecu 322
Zadatak 37 – Grupisanje zaposlenih 323
Zadatak 38 – Prodaja za svaki mesec unutar godine 324
Zadatak 39 – Razlika u iznosu prodaje 325
Zadatak 40 – Godišnja razlika u ukupnoj prodaji po klijentima 326
Zadatak 41 – Rekurzivni upit za kreiranje hijerarhijske strukture zaposlenih 327
Zadatak 42 – Prozorske funkcije 329
Zadatak 43 – Prilagođavanje i poboljšanje strukture tabele 330
Zadatak 44 – Rešavanje problema sa kolonom 331
Zadatak 45 – Rešavanje problema sa viševrednosnim kolonama 332
Zadatak 46 – Osiguranje dobro definisane i stabilne strukture baze podataka 333
Zadatak 47 – Problem nepotrebnih dupliranih kolona 334
Zadatak 48 – Osiguranje jasne identifikacije svakog reda 335
Zadatak 49 – Implementiranje upita 336
Zadatak 50 – Implementiranje upita za identifikovanje zaposlenih 337
Zadatak 51 – Pretraga imena zaposlenih 338
Zadatak 52 – Zajednički elementi dva skupa rezultata 339
Zadatak 52 – Redovi jedne tabele povezani sa drugom tabelom 340
Zadatak 53 – Proizvodi koji nikad nisu naručeni 341
Zadatak 54 – Svi kontakti koji odgovaraju određenom kriterijumu 342
Zadatak 55 – Podupiti za zaposlene koji imaju platu veću od prosečne plate 343
Zadatak 56 -Podupiti za proizvode kojima je cena niža od prosečne 344
Zadatak 57 – Skalarni podupit za izračunavanje razlike u plati 345
Zadatak 58 – Podupit za izračuvanje broja dana 345
Zadatak 59 – Podupit kao izraz kolona 346
Zadatak 60 – Podupit kao filter za identifikaciju zaposlenih 347
Zadatak 61 – Generisanje sledeće vrednosti primarnog ključa 348
Zadatak 62 – Nevezani podaci 349
Zadatak 63 – Referentna tabela za rešavanje problema 350
Zadatak 64 – Identifikacija kupaca koji su izvršili više od jedne kupovine 351
Zadatak 65 – Identifikacija potencijalnih grešaka u unosu podataka 352
Zadatak 66 – Rešavanje problema redundacije podataka 353
Zadatak 67 – Kontrola zaključavanja za optimizaciju performansi 354
Zadatak 68 - Rešavanje konflikta između konkurentnih operacija 355
Zadatak 69 – Dodeljivanje privilegija za pristup i manipulaciju podacima 356
Zadatak 70 – Dizajniranje „od vrha do dna" baze podataka 357

POGLAVLJE 71

Pitanja za intervju za posao 358

Pitanja za intervju za posao, kao osnova za dodatnu obuku – srednji nivo....358
Objasnite razliku između klauzula INNER JOIN, LEFT JOIN, RIGHT JOIN i FULL OUTER JOIN u SQL jeziku. 358
Kako možete da optimizujete performanse SQL upita?....360
Šta je transakcija u SQL jeziku i kako biste implementirali kontrolu transakcija?....361
Možete li objasniti kako funkcioniše indeksiranje i kako utiče na performanse SQL upita?....363
Kako biste identifikovali i rešili problem „N+1" u SQL upitima?....365
Kako biste projektovali i implementirali rekurzivni upit u SQL jeziku?....367
Objasnite razliku između klauzula HAVING i WHERE u SQL jeziku.....368
Kako biste implementirali sigurnost na nivou podataka u SQL bazi podataka?....370

POGLAVLJE 72

Pomoćne tabele (CTE) u SQL programskom jeziku. 372

Uvod....372
Pomoćna tabela - Rekurzivni upiti....373
Pomoćna tabela - složene analize i obrade....377
Pomoćna tabela - zamena za privremene tabele....382
Pomoćna tabela - obrada podskupova podataka....386
Pomoćna tabela - složeno filtriranje i sortiranje....388
Pomoćna tabela – Izrada prototipa i testiranje....390
Pomoćna tabela - Obrada geoprostornih podataka....391
Pomoćna tabela - Složeno rangiranje....392
Pomoćna tabela - priprema podataka za mašinsko učenje....393
Pomoćna tabela - Analiza trendova....394

POGLAVLJE 73

SQLgpt odgovori na pitanja kojih nema u knjizi 396

Objasnjenje rešenja:....397
Performanse i optimizacija:....397
Ograničenja i zablude:....397
Saveti i najbolja praksa:....397
Zaključak:....398
Kviz:....398
Pojam ili problem:....398
Kreiranje upita i rešenje:....398
Primer upita:....399

ZAŠTO JE KORISNO UČITI SQL?

Učenje programskog jezika SQL (Structured Query Language) je korisno iz nekoliko razloga:

1. **Univerzalni jezik za upravljanje bazama podataka**: SQL je standardni jezik koji se koristi za pretragu, ažuriranje, i manipulaciju podacima u relacionim bazama podataka. Znanje programskog jezika SQL omogućava vam da radite sa različitim sistemima za obradu baza podataka kao što su MySQL, PostgreSQL, SQL Server, Oracle, i druge.
2. **Važnost u analizi podataka**: U današnjem svetu, gde su podaci izuzetno vredni, SQL vam omogućava da brzo i efikasno izvlačite i analizirate podatke, što je ključno u mnogim industrijama za donošenje informisanih odluka.
3. **Fleksibilnost i svestranost**: SQL se može koristiti u širokom spektru aplikacija - od jednostavnih upita do složene analize podataka, izveštavanja, i automatskog generisanja podataka.
4. **Tražena veština na tržištu rada**: Znanje programskog jezika SQL je izuzetno cenjeno u mnogim oblastima IT industrije kao što su razvoj softvera, analiza podataka, upravljanje bazama podataka, inženjering podataka, i poslovna inteligencija.
5. **Osnova za naprednije tehnologije**: Znanje programskog jezika SQL često služi kao temelj za učenje naprednijih tehnologija i koncepata kao što su veliki skupovi podataka (big data), istraživanje podataka (data mining), i mašinsko učenje (machine learning).
6. **Podrška za donošenje odluka i izveštavanje**: SQL omogućava korisnicima da stvaraju i koriste izveštaje koji mogu pomoći u nadzoru performansi i donošenju poslovnih odluka.
7. **Efikasnost u radu s podacima**: SQL omogućava brzo i efikasno upravljanje velikim količinama podataka, što je neophodno u mnogim modernim aplikacijama.

Učenje programskog jezika SQL nudi duboko razumevanje kako se podaci skladište, pretražuju i kako se njima manipuliše, što je ključno za bilo koga ko želi da radi u oblastima gde su podaci od suštinskog značaja.

Univerzalni jezik za upravljanje bazama podataka

Programski jezik SQL, kao standardizovani jezik za upravljanje relacionim bazama podataka, ima nekoliko ključnih prednosti koje ga čine univerzalnim alatom u svetu podataka:

- **Široka primena**: SQL se koristi u različitim sistemima za upravljanje bazama podataka (DBMS), kao što su MySQL, PostgreSQL, Oracle, Microsoft SQL Server, i mnogi drugi. Ovo znači da, jednom kada naučite jezik SQL, možete raditi sa skoro bilo kojom relacionom bazom podataka.
- **Standardizovan, ali prilagodljiv**: Iako postoji standardni SQL (ANSI SQL) jezik, mnogi sistemi za upravljanje bazama podataka imaju svoje specifične dodatke ili varijacije. Međutim, osnovni koncepti i većina sintakse ostaju konzistentni, što omogućava lak prelazak između različitih sistema.
- **Neprolaznost i trajna relevantnost**: SQL postoji i koristi se već decenijama. Iako se tehnologije razvijaju i vremenom menjaju, potreba za efikasnim upravljanjem podacima u relacionim bazama podataka ostaje, čineći poznavanje SQL jezika ključnom veštinom u mnogim industrijskim oblastima.
- **Integracija sa drugim tehnologijama**: SQL se lako integriše s drugim programskim jezicima i alatima, omogućavajući razvoj složenih aplikacija i sistema koji zahtevaju rad sa podacima.

U suštini, SQL je jezik koji vam omogućava da „razgovarate" sa bazama podataka. Bez obzira na specifični sistem za upravljanje bazama podataka koji koristite, osnovne komande kao što su SELECT, INSERT, UPDATE, DELETE, i JOIN su univerzalne, i znanje SQL jezika vam omogućava da pristupate, analizirate, i manipulišete podacima unutar bilo koje relacione baze podataka.

Važnost u analizi podataka

SQL je ključan alat za analizu podataka iz nekoliko razloga:

- **Pristupanje i izvlačenje podataka**: SQL omogućava korisnicima da efikasno pretražuju i izvlače podatke iz baza podatka na osnovu različitih jednostavnih i složenih kriterijuma. Ovo je posebno važno u okruženjima gde su podaci složeni i obimni. Na primer, koristeći SQL, možete lako filtrirati podatke, izvršavati složene upite, i izvlačiti tačne informacije potrebne za analizu.
- **Manipulacija podacima**: SQL omogućava transformaciju podataka u željeni format. Možete koristiti različite funkcije (kao što su agregatne funkcije poput SUM i AVG) i klauzule (kao što su GROUP BY i ORDER BY) za pripremu podataka za analizu.
- **Analiza velikih skupova podataka**: U današnjem dobu velikih skupova podataka, SQL se pokazuje kao izuzetno efikasan u obradi i analizi takvih skupova podataka. Većina modernih sistema za upravljanje bazama podataka optimizovana je za brzo izvršavanje složenih SQL upita.
- **Osnova za poslovnu inteligenciju (BI) i izveštavanje**: SQL je često korišćen za izradu izveštaja i kontrolnih tabli u alatima za poslovnu inteligenciju. Znanje SQL jezika omogućava vam da prilagodite izveštaje i analize specifičnim potrebama organizacije.

- **Podrška za donošenje odluka**: Precizna i brza analiza podataka je ključna za informisano donošenje odluka u poslovanju. SQL omogućava brzo pristupanje relevantnim podacima, što menadžerima i analitičarima pomaže da razumeju trenutne performanse i predvide buduće trendove.

Učenje SQL programskog jezika je stoga neophodno za svakoga ko se bavi analizom podataka, bilo da je to u kontekstu poslovne analize, nauke o podacima, marketinga, finansija ili bilo koje druge oblasti gde su podaci od suštinskog značaja za donošenje odluka.

Fleksibilnost i svestranost

SQL nudi široku paletu mogućnosti koje ga čine fleksibilnim i svestranim alatom u radu sa bazama podataka:

- **Kompatibilnost sa različitim tipovima podataka**: SQL može da rukuje različitim tipovima podataka, uključujući tekstualne, numeričke, datumsko-vremenske i binarne tipove podataka. Ovo ga čini pogodnim za različite aplikacije i scenarije upotrebe.
- **Raznolikost operacija**: SQL podržava širok spektar operacija nad podacima, uključujući pretragu, unos, ažuriranje, i brisanje podataka, kao i složenije operacije poput spajanja (joins) podataka iz različitih tabela, grupisanja, i sortiranja rezultata.
- **Prilagodljivost upitima**: Korisnici mogu da prilagode SQL upite da odgovaraju specifičnim potrebama analize ili izveštavanja. SQL upiti mogu biti jednostavni (npr. izvlačenje podataka iz jedne tabele) ili složeni (npr. korišćenje podupita ili funkcija za agregiranje).
- **Skalabilnost**: SQL se efikasno koristi kako u malim, tako i u velikim sistemima baza podataka. Može da obradi velike količine podataka, što ga čini idealnim za rad u okruženjima koja zahtevaju obradu velikih količina podataka.
- **Primena u različitim industrijama**: SQL se koristi u širokom rasponu industrija, od tehnologije i finansija do zdravstva i obrazovanja. Njegova univerzalnost ga čini važnim alatom za skoro svaku organizaciju koja radi sa podacima.
- **Integracija sa drugim tehnologijama**: SQL se lako integriše sa drugim tehnologijama i alatima, uključujući aplikacije za izveštavanje, alate za analizu podataka, i programske jezike kao što su Python i Java. Ovo omogućava automatizaciju i optimizaciju procesa rada sa podacima.

Zbog ovih karakteristika, SQL se smatra fleksibilnim i svestranim jezikom koji je neophodan za profesionalce u svetu podataka, bez obzira na specifičnu oblast ili industriju u kojoj rade.

Tražena veština na tržištu rada

Poznavanje SQL jezika je izuzetno tražena veština na tržištu rada iz više razloga:

- **Široka primena u različitim industrijama**: SQL se koristi u različitim sektorima, uključujući IT industriju, bankarstvo, osiguranje, zdravstvo, maloprodaju, i mnoge druge. Skoro svaka industrija koja se oslanja na sakupljanje i analizu podataka zahteva stručnjake koji znaju SQL.

- **Osnova za poslovne karijere u radu sa podacima**: Za mnoge poslove povezane sa radom sa podacima, kao što su analiza podataka, nauka o podacima, razvoj baza podataka, i poslovna inteligencija, znanje SQL jezika je ključno. Čak i u ulogama koje nisu direktno povezane s podacima, kao što su razvoj softvera, upravljanje projektima, i poslovna savetovanja, SQL može biti koristan alat.
- **Potreba za efikasnom obradom i analizom podataka**: U današnjem poslovnom svetu, sposobnost brze i precizne obrade i analize podataka je ključna. Poznavanje rada sa SQL jezikom omogućava profesionalcima da efikasno izvlače i manipulišu podacima, što je važno za donošenje poslovnih odluka.
- **Prednost u zapošljavanju i napredovanju u karijeri**: Kandidati koji znaju SQL često imaju prednost pri zapošljavanju za mnoge tehničke i analitičke uloge. Takođe, znanje SQL jezika može pomoći postojećim zaposlenima da napreduju u svojim karijerama, preuzimajući složenije projekte ili vodeće uloge.
- **Prilagođavanje digitalnoj transformaciji**: Kako se organizacije sve više oslanjaju na digitalne tehnologije, sposobnost upravljanja i analize podataka postaje sve važnija. SQL je temeljni alat u ovom procesu i zaposleni koji su vešti u radu sa SQL jezikom su ključni za uspešnu digitalnu transformaciju.

Ukratko, poznavanje SQL jezika je trajna i univerzalno tražena veština koja otvara vrata različitim karijernim mogućnostima i pomaže profesionalcima da se istaknu u dinamičnom i konkurentnom poslovnom okruženju.

Osnova za naprednije tehnologije

SQL se ne koristi samo za trenutne potrebe u upravljanju podacima, već služi i kao temelj za usvajanje i razumevanje naprednijih tehnologija i koncepata:

- **Veliki skupovi podataka**: U kontekstu velikih skupova podataka, koji su sveprisutni u savremenom poslovnom svetu, SQL omogućava obradu i analizu ogromnih količina informacija. Tehnologije poput Apache Hadoop i Spark često koriste SQL ili SQL slične jezike za efikasno upravljanje podacima.
- **Nauka o podacima i mašinsko učenje**: Za analitičare i naučnike u oblasti podataka, SQL je alat za pripremu i čišćenje podataka pre primene metoda mašinskog učenja. Mnogi alati u domenu nauke o podacima omogućavaju integraciju sa SQL bazama podataka.
- **Računarstvo u oblaku**: Sa rastom tehnologija računarstva u oblaku, SQL nalazi svoju primenu u uslugama baziranim na oblaku, koje nude Amazon, Microsoft Azure, i Google Cloud, omogućavajući efikasno skladištenje i obradu podataka u oblaku.
- **Razvoj veb aplikacija**: U razvoju veb aplikacija, SQL se koristi za interakciju sa bazama podataka na serveru. Takođe, razvojni programeri sa punim skupom veština često koriste SQL za upravljanje delom aplikacije na strani servera.
- **Poslovna inteligencija i analitika**: Alati za poslovnu inteligenciju kao što su Tableau, Power BI i Qlik koriste SQL za izvlačenje podataka iz različitih izvora, omogućavajući kreiranje detaljnih izveštaja i vizuelizacija.
- **Automatizacija i izveštavanje**: SQL pomaže u automatizaciji redovnih zadataka upravljanja podacima i generisanju izveštaja, što je od suštinskog značaja za mnoge poslovne procese.

Zaključak je da, dok tehnologije napreduju, SQL zadržava svoju poziciju kao osnovni alat u svetu podataka, pružajući stabilnu platformu za razvoj veština i znanja koji su neophodni za rad sa savremenim tehnološkim rešenjima.

Podrška za donošenje odluka i izveštavanje

SQL igra ključnu ulogu u podršci procesu donošenja odluka i izveštavanju u organizacijama iz različitih razloga:

- **Brz i precizan pristup podacima**: SQL omogućava korisnicima da brzo i tačno pristupe podacima koji su im potrebni. Ovo je posebno važno u poslovanju gde se odluke moraju donositi brzo i na osnovu tačnih informacija.
- **Analiza podataka**: SQL se koristi za analizu podataka, omogućavajući korisnicima da identifikuju trendove, uzorke, i anomalije. Ovo pomaže u donošenju odluka zasnovanih na podacima.
- **Izveštavanje i vizuelizacija**: SQL se često koristi za izradu izveštaja i vizuelizacije podataka. Ovo omogućava menadžerima i drugim rukovodiocima da lako razumeju složene podatke i informacije.
- **Personalizovano izveštavanje**: Koristeći SQL, analitičari mogu kreirati prilagođene izveštaje koji odgovaraju specifičnim potrebama korisnika ili odseka unutar organizacije.
- **Automatizacija procesa izveštavanja**: SQL omogućava automatizaciju redovnih izveštajnih zadataka, smanjujući vreme potrebno za pripremu izveštaja i smanjujući mogućnost ljudske greške.
- **Osnova za poslovnu inteligenciju**: Alati za poslovnu inteligenciju često koriste SQL za izvlačenje i analizu podataka, pružajući korisnicima moćne alate za vizuelizaciju i razumevanje podataka.

Ukratko, znanje SQL jezika omogućava profesionalcima da efikasno upravljaju podacima, što direktno doprinosi boljem razumevanju poslovnih operacija, povećanju efikasnosti, i donošenju boljih poslovnih odluka.

Efikasnost u radu s podacima

SQL poboljšava efikasnost u radu s podacima kroz nekoliko ključnih aspekata:

- **Brzi pristup i obrada podataka**: SQL omogućava korisnicima da brzo pristupe i obrade velike količine podataka. Upiti mogu biti napisani i izvršeni u kratkom vremenskom periodu, što omogućava efikasnu obradu podataka.
- **Složeni upiti i analize**: SQL podržava složene upite, uključujući spajanje (join) više tabela, filtriranje podataka pomoću uslovnih izraza, grupisanje podataka, i primenu agregatnih funkcija. Ovo omogućava korisnicima da izvrše detaljne analize i dobiju uvide iz podataka.
- **Standardizacija i automatizacija**: SQL je standardizovan jezik, što znači da se isti ili slični upiti mogu koristiti u različitim sistemima bazama podataka. Ovo olakšava automatizaciju redovnih zadataka, kao što su izveštavanje, ažuriranje podataka, i održavanje baza.

- **Optimizacija performansi**: SQL omogućava analizu i optimizaciju performansi upita, što je ključno za rad s velikim bazama podataka. Indeksiranje, planiranje upita, i druge tehnike optimizacije mogu se koristiti za poboljšanje brzine i efikasnosti upita.
- **Skalabilnost**: SQL se može koristiti kako u malim, tako i u velikim bazama podataka. Njegova skalabilnost omogućava organizacijama da se prilagode rastućim potrebama za obradom podataka.
- **Integracija sa drugim sistemima**: SQL se lako integriše sa drugim aplikacijama i sistemima, što omogućava razmenu podataka i koordinaciju između različitih tehnoloških sistema unutar organizacije.

Zbog ovih karakteristika, SQL značajno doprinosi efikasnosti u radu s podacima, omogućavajući brzo, precizno, i fleksibilno upravljanje podacima. Ovo je ključno u današnjem poslovnom svetu gde se brza i tačna analiza podataka smatra vitalnom za uspeh.

Budućnost SQL jezika

Budućnost SQL jezika u svetu tehnologije se smatra veoma svetlom iz više bitnih razloga, a evo detaljnijeg objašnjenja svakog od njih:

1. **Neprocenjiva uloga u radu s podacima**: Bez obzira na napredak tehnologija, SQL zadržava svoju osnovnu ulogu u svetu relacionih baza podataka. Sposobnost SQL jezika da precizno, brzo i efikasno upravlja, pretražuje i obrađuje podatke čini ga nezamenljivim u mnogim aplikacijama i industrijama. SQL je kao temelj koji podržava zgradu podataka - možda ne uvek vidljiv, ali apsolutno neophodan za stabilnost i funkcionalnost.
2. **Usklađivanje sa savremenim tehnološkim trendovima**: SQL se neprekidno razvija i prilagođava kako bi bio u koraku s najnovijim tehnološkim dostignućima. Njegova integracija sa tehnologijama računarstva u oblaku, platformama za obradu velikih podataka kao što su Hadoop i Spark, i alatima za analitiku i poslovnu inteligenciju pokazuje njegovu fleksibilnost i prilagodljivost modernom tehnološkom okruženju.
3. **Važnost u svetu tehnologija koje rade sa velikim skupovima podataka**: SQL postaje sve važniji igrač u domenu tehnologija koje rade sa velikim skupovima podataka, koristeći SQL slične jezike poput Apache HiveQL. To olakšava analitičarima, koji su već upoznati sa SQL jezikom, da pređu na obradu i analizu ogromnih skupova podataka, što je ključno u današnjem svetu vođenom podacima.
4. **Razvoj i širenje NoSQL i NewSQL baza podataka**: Dok NoSQL baze pružaju fleksibilnost za specifične vrste podataka, mnoge od njih počinju da uključuju SQL slične jezike, pokazujući važnost SQL jezika. Takođe, razvoj NewSQL baza podataka spaja performanse NoSQLjezika sa pouzdanošću i lakoćom upotrebe SQL jezika, što predstavlja najbolje iz oba sveta upitnih jezika.
5. **Uloga u automatizaciji i veštačkoj inteligenciji**: U doba veštačke inteligencije i mašinskog učenja, SQL nalazi svoje mesto u automatizaciji složenih procesa obrade podataka. Očekuje se da će SQL imati centralnu ulogu u budućem razvoju i integraciji veštačke inteligencije i analitičkih platformi.

6. **Obrazovanje i stručno usavršavanje**: SQL ostaje ključna stavka u obrazovanju budućih generacija programera, analitičara i naučnika koji se bave obradom podataka. Njegovo mesto u nastavnim programima obrazovnih ustanova je čvrsto i neophodno za formiranje temelja u razumevanju i upravljanju podacima.
7. **Podrška zajednice**: Velika i aktivna zajednica korisnika SQL jezika neprekidno pruža podršku, resurse i inovacije, što doprinosi dugovečnosti i održivosti SQL jezika kao tehnologije.

Zaključno, iako se tehnološki svet neprestano menja i razvija, SQL ostaje ključan alat u arsenalu svakog stručnjaka koji radi sa podacima. Njegova sposobnost da se razvija i prilagodi se novim izazovima obezbeđuje mu trajnu relevantnost u dinamičnom tehnološkom svetu.

7

SELECT NAREDBA

Uvod i sintaksa

Naredba **SELECT** je osnovna naredba u SQL jeziku koja se koristi za preuzimanje podataka iz baze podataka. Sintaksa naredbe **SELECT** glasi:

```
SELECT kolona1, kolona2, ... FROM tabela;
```

gde `kolona1`, `kolona2`, **...** predstavljaju imena kolona koje želite da izaberete, a `tabela` je ime tabele iz koje preuzimate podatke.

Kako funkcioniše

Naredba **SELECT** pretražuje bazu podataka i vraća skup podataka (obično u obliku tabele) koji zadovoljava kriterijume navedene u upitu. Može da se koristi za preuzimanje podataka, bilo da je reč o jednoj koloni ili kompletnim redovima, iz jedne ili više tabela. Naredbom **SELECT** mogu da se preuzmu i podaci iz svih kolona iz jedne ili više tabela. Tada umesto navođenja imena svih kolona dovoljno je koristiti * koja je zamena za imena kolona.

Primeri upotrebe

Osnovni nivo

Preuzimanje imena i prezimena studenata iz tabele studenti:

```
SELECT ime, prezime FROM studenti;
```

Ovo je jednostavan SQL upit koji će izabrati imena i prezimena svih studenata iz tabele studenti. Ova naredba se koristi za dobijanje informacija samo o kolonama `ime` i `prezime` iz tabele `studenti` bez dodatnih uslova ili ograničenja.

Srednji nivo

Preuzimanje podataka o studentima koji su na trećoj godini studijauz uslov da se podaci sortiraju po prezimenu:

```
SELECT ime, prezime FROM studenti WHERE godina_studija = 3 ORDER BY prezime;
```

U ovom malo složenijem SQL upitu postavljen je uslov. Samo redovi gde je vrednost u koloni `godina_studija` jednaka `3` će biti uključeni u rezultate. Pored toga, ključna reč **ORDER BY** se koristi za sortiranje rezultata prema vrednostima u koloni `prezime`. Podaci će biti prikazani u rastućem redosledu (podrazumevano), što znači da će rezultati biti prikazani u abecednom redosledu prezimena.

Napredni nivo

Preuzimanje ukupnog broja studenata po godini studija:

```
SELECT godina_studija, COUNT(*) AS broj_studenata FROM studenti GROUP BY
godina_studija;
```

Ovaj SQL upit predstavlja agregatni upit. Rezultat **COUNT(*)** se koristi u kombinaciji sa naredbom **GROUP BY**. Navođenjem kolone `godina_studija` u okviru ove naredbe, grupišu se rezultati prema vrednostima te kolone. To znači da se **COUNT(*)** primenjuje unutar svake grupe, tako da dobijate broj studenata po svakoj godini studija. Klauzula **AS** `broj_studenata` se koristi za dodelu imena rezultatima brojanja.

Performanse

Performanse naredbe **SELECT** mogu varirati zavisno od veličine i strukture baze podataka, kao i od složenosti samog upita. Što je tabela veća, to će pretraga biti sporija. Kada se izvršava **SELECT** upit, baza podataka mora proći kroz sve redove kako bi pronašla podatke koji ispunjavaju uslove upita. Indeksiranje kolona koje se često koriste u **WHERE** klauzulama može znatno poboljšati brzinu pretrage. Pravilno postavljeni indeksi mogu biti od suštinskog značaja, posebno u slučajevima kada imate veliku količinu podataka. Međutim, treba imati na umu da indeksiranje takođe može uticati na performanse upisa i ažuriranja, jer svaka promena podataka u indeksiranoj koloni zahteva i ažuriranje indeksa.

Složenost **SELECT** upita, odnosno način na koji je napisan i vrsta operacija koje zahteva, može značajno uticati na performanse. Nekoliko faktora koje treba uzeti u obzir su broj operacija i funkcija, join operacije, podupiti, sortiranje i grupisanje, korišćenje funkcija u **WHERE** klauzuli itd.

Ograničenja i zablude

Najveće ograničenje naredbe **SELECT** je da može dovesti do velikog opterećenja na serveru ako se ne koristi pažljivo, posebno na velikim bazama podataka. Zabluda je da je **SELECT *** uvek najbolja opcija; često je efikasnije izabrati samo potrebne kolone. Korišćenje **SELECT *** vraća sve kolone iz tabele, što može dovesti do slanja velike količine podataka iz baze podataka na aplikaciju. Ovo može uzrokovati značajan gubitak performansi, naročito kada se radi s velikim skupovima podataka ili kada mrežni resursi nisu optimalni. Preporučljivo je specifično navesti samo one kolone koje su zaista potrebne za trenutni kontekst ili operaciju. Ovo smanjuje količinu podataka koja se prenosi iz baze podataka, ubrzava brzinu izvršavanja upita i smanjuje opterećenje servera.



Saveti i najbolja praksa:

Koristite specifična imena kolona umesto SELECT *. Umesto da koristite SELECT *, navedite tačno koje kolone želite da izaberete. Ovo smanjuje količinu prenesenih podataka i poboljšava performanse upita.

Koristite WHERE klauzulu za filtriranje podataka. Upotreba WHERE klauzule za filtriranje podataka smanjuje broj redova koji se obrađuju. Dodatno, ako indeksirate često korišćene kolone u WHERE klauzulama, ubrzaćete pretragu i optimizaciju upita.

Primena indeksiranja na često korišćene kolone u pretragama. Ako koristite JOIN operacije za spajanje podataka iz više tabela, proverite da li su indeksirane kolone koje koristite za spajanje. Pravilno indeksirane kolone mogu znatno poboljšati brzinu JOIN operacija.

Ograničite broj redova koji se vraćaju koristeći LIMIT. Kada očekujete veliki broj rezultata, koristite LIMIT kako biste ograničili broj vraćenih redova. Ovo je posebno korisno u situacijama gde želite videti samo prvih nekoliko rezultata.

Razmatrajte upotrebu agregatnih funkcija. Agregatne funkcije poput SUM, AVG, MIN, i MAX mogu biti korisne, ali budite svesni njihovog uticaja na performanse, posebno kada se koriste na velikim skupovima podataka.

Optimizujte korišćenje podupita. Ako koristite podupite, proverite da li su efikasni i ne izvršavaju se više puta nego što je potrebno. Nepotrebni podupiti mogu dodati složenost i smanjiti performanse.

Zaključak

Naredba **SELECT** je moćan alat za preuzimanje podataka iz baza podataka, a njenim razumevanjem se ključno doprinosi efikasnom upravljanju i analizi podataka.

Kviz

Pitanje: Da li naredba **SELECT * FROM studenti;** vraća podatke o svim studentima iz baze podataka?

- **Tačno:** Da, vraća podatke o svim studentima iz tabele **studenti**.
- **Netačno:** Ne, vraća samo imena studenata.

Objašnjenje: Tačan odgovor je da **SELECT * FROM studenti;** vraća sve kolone i sve redove iz tabele **studenti**, što znači da dobijamo kompletne informacije o svakom studentu. Netačan odgovor pogrešno pretpostavlja da se vraćaju samo imena.

42

FOREIGN KEY OGRANIČENJE

Uvod i sintaksa

Ograničenje **FOREIGN KEY** u SQL jeziku ključno je za održavanje referencijalnog integriteta među tabelama unutar relacione baze podataka. Ovo ograničenje omogućava da kolone u jednoj tabeli referenciraju primarne ključeve ili jedinstvene ključeve druge tabele, uspostavljajući tako vezu između dve tabele.

Kada se **FOREIGN KEY** ograničenje primeni, vrednosti u koloni koja je označena kao strani ključ moraju postojati kao vrednosti primarnog ili jedinstvenog ključa u referentnoj tabeli. Sintaksa za definisanje **FOREIGN KEY** ograničenja je sledeća:

```
CREATE TABLE ImeTabele (
    Kolona1 TipPodatka,
    Kolona2 TipPodatka,
    ...
    FOREIGN KEY (Kolona1) REFERENCES DrugaTabela(PrimarniKljuc),
    ...
);
```

Deo SQL upit **FOREIGN KEY (Kolona1) REFERENCES DrugaTabela (PrimarniKljuc)** definiše strani ključ u tabeli. Evo šta svaki deo znači: **FOREIGN KEY (Kolona1)** definiše strani ključ u tabeli **ImeTabele** koji referencira kolonu **Kolona1**. **REFERENCES DrugaTabela(PrimarniKljuc)** navodi da se strani ključ odnosi na tabelu **DrugaTabela** i njen primarni ključ **PrimarniKljuc**. Ova konstrukcija uspostavlja vezu između polja u trenutnoj tabeli i primarnog ključa u drugoj tabeli, čime se postiže referencijalna integritet.

Kako funkcioniše

Kada se doda ili ažurira red u tabeli sa **FOREIGN KEY** ograničenjem, baza podataka proverava da li vrednosti stranog ključa postoje u referentnoj tabeli. Ako referentna vrednost ne postoji, operacija dodavanja ili ažuriranja biće odbijena, čime se osigurava referencijalni integritet.

Primeri upotrebe

Osnovni nivo

Povezivanje tabele **Narudzbine** sa tabelom **Kupci** preko stranog ključa **KupacID**:

```
CREATE TABLE Narudzbine (
    NarudzbinaID int PRIMARY KEY,
    KupacID int,
    FOREIGN KEY (KupacID) REFERENCES Kupci(KupacID)
);
```

Ovaj primer osigurava da svaka narudžbina mora imati ispravno definisanog kupca.

Srednji nivo

Kreiranje više **FOREIGN KEY** ograničenja unutar iste tabele:

```
CREATE TABLE NarudzbineDetalji (
    NarudzbinaDetaljID int PRIMARY KEY,
    NarudzbinaID int,
    ProizvodID int,
    FOREIGN KEY (NarudzbinaID) REFERENCES Narudzbine(NarudzbinaID),
    FOREIGN KEY (ProizvodID) REFERENCES Proizvodi(ProizvodID)
);
```

Ovaj primer uspostavlja veze sa tabelama **Narudzbine** i **Proizvodi**, osiguravajući ispravnost narudžbine i proizvoda u svakom detalju narudžbine.

Napredni nivo

Kaskadno brisanje i ažuriranje pomoću **FOREIGN KEY** ograničenja:

```
CREATE TABLE NarudzbineDetalji (
    NarudzbinaDetaljID int PRIMARY KEY,
    NarudzbinaID int,
    FOREIGN KEY (NarudzbinaID) REFERENCES Narudzbine(NarudzbinaID)
    ON DELETE CASCADE
);
```

Ovaj primer omogućava automatsko brisanje detalja narudžbine kada se odgovarajuća narudžbina obriše, održavajući tako integritet podataka.

Performanse

FOREIGN KEY ograničenje može uticati na performanse, posebno prilikom upisivanja ili ažuriranja podataka. Kada se izvršava ovakva operacija, sistem mora da proveri da li postoji odgovarajući zapis u referenciranoj tabeli, što može dovesti do dodatnih koraka i vremena izvršavanja. Međutim, koristi u smislu očuvanja tačnosti podataka obično nadmašuju potencijalne troškove performansi. U mnogim implementacijama, kreiranje indeksa na poljima koja se koriste kao FOREIGN KEY može poboljšati performanse provere referencijalnog integriteta. Međutim, indeksiranje takođe ima svoje troškove, posebno prilikom ažuriranja i brisanja podataka.

Ograničenja i zablude

Jedna od zabluda može biti da **FOREIGN KEY** ograničenje značajno usporava sve operacije baze podataka. Iako postoji određeni uticaj na operacije upisa, moderni sistemi za upravljanje bazama podataka su optimizovani da minimiziraju taj uticaj. Sistemi za upravljanje bazama podataka često koriste različite tehnike, uključujući pametno keširanje podataka, indeksiranje i optimizacije upita kako bi ublažili potencijalno usporenje koje proizlazi iz provere referencijalnog integriteta. Takođe, treba imati na umu da nepravilno korišćenje **FOREIGN KEY** ograničenja može dovesti do kompleksnih zavisnosti između tabela, što može otežati održavanje baze podataka, posebno ako se menjaju struktura podataka ili ako je potrebno vršiti migracije podataka.



Saveti i najbolja praksa:

Pažljivo planirajte referencijalne veze između tabela kako biste izbegli nepotrebno složene zavisnosti.

Razmotrite korišćenje kaskadnih operacija (ON DELETE CASCADE, ON UPDATE CASCADE) sa oprezom, jer mogu dovesti do neželjenog brisanja ili ažuriranja velikog broja redova.

Ako želite više kontrole nad kaskadnim operacijama, razmotrite korišćenje okidača umesto direktnog korišćenja kaskadnih operacija. Ovo omogućava dodatnu fleksibilnost i jasnije definisanje akcija koje se preduzimaju na promeni podataka.

Koristite FOREIGN KEY ograničenja za očuvanje referencijalnog integriteta i jasno definisanje veza između entiteta u vašem modelu podataka.

Zaključak

FOREIGN KEY ograničenje je moćan mehanizam za očuvanje referencijalnog integriteta i definisanje odnosa između tabela u relacionoj bazi podataka. Pravilnom upotrebom, **FOREIGN KEY** ograničenja doprinose strukturi, tačnosti i pouzdanosti podataka unutar baze podataka.

Kviz

Pitanje: Da li **FOREIGN KEY** ograničenje dozvoljava **NULL** vrednosti u stranom ključu?

- **Tačno:** Da, FOREIGN KEY ograničenje dozvoljava NULL vrednosti u stranom ključu, pod uslovom da to nije eksplicitno zabranjeno drugim ograničenjima.
- **Netačno:** Ne, FOREIGN KEY ograničenje ne dozvoljava NULL vrednosti u stranom ključu.

Objašnjenje: **FOREIGN KEY** ograničenje dozvoljava **NULL** vrednosti u stranom ključu, što omogućava postojanje redova koji trenutno nemaju referencu na red u referentnoj tabeli. Ovo može biti korisno u situacijama gde veza između redova nije odmah poznata ili primenljiva.

DROP FOREIGN KEY ograničenje

Uklanjanje (**DROP**) ograničenja **FOREIGN KEY** iz tabele u SQL jeziku koristi se kada je potrebno eliminisati postojeću vezu između dve tabele. Ova operacija se može izvesti kada želite da promenite strukturu baze podataka, uklonite nepotrebne veze ili pripremite tabelu za nove veze.

Uvod i sintaksa

Da biste uklonili **FOREIGN KEY** ograničenje, koristi se **ALTER TABLE** naredba zajedno sa **DROP CONSTRAINT** klauzulom. Važno je znati ime ograničenja **FOREIGN KEY**, koje se može pronaći u definiciji tabele ili korišćenjem alata za upravljanje bazom podataka. Sintaksa za uklanjanje stranog ključa je:

```
ALTER TABLE ImeTabele
DROP CONSTRAINT ImeStranogKljucOgranicenja;
```

Kako funkcioniše

Kada se izvrši naredba za uklanjanje **FOREIGN KEY** ograničenja, SQL sistem uklanja vezu između dve tabele definisanu tim ograničenjem. Ovo znači da nakon uklanjanja, kolone koje su bile pod uticajem **FOREIGN KEY** ograničenja više neće imati restrikcije povezivanja sa referentnom tabelom.

Zadatak 57 – Skalarni podupit za izračunavanje razlike u plati

*Kako biste konstruisali SQL upit koji koristi skalarne podupite za izračunavanje razlike u plati između svakog zaposlenog i najviše plate u kompaniji, uzimajući u obzir tabelu **Zaposleni** koja sadrži kolone za šifru zaposlenog, ime, prezime, i platu, i kako biste osigurali da se u rezultatima prikažu šifra zaposlenog, ime, prezime, plata, kao i izračunata razlika u plati?*

SQLgpt: Da bismo konstruisali SQL upit koji koristi skalarne podupite za izračunavanje razlike u plati između svakog zaposlenog i najviše plate u kompaniji, možemo koristiti podupit unutar SELECT klauzule:

```
SELECT
    ID_zaposlenog,
    Ime,
    Prezime,
    Plata,
    (SELECT MAX(Plata) FROM Zaposleni) - Plata
      AS RazlikaUPlati
FROM
    Zaposleni;
```

Objašnjenje upita:

1. **Selekcija osnovnih informacija zaposlenih**: Osnovni deo upita selektuje šifru zaposlenog, ime, prezime i platu svakog zaposlenog iz tabele **Zaposleni**.
2. **Skalarni podupit za najvišu platu**: Unutar **SELECT** klauzule, koristi se skalaran podupit **(SELECT MAX(Plata) FROM Zaposleni)** da bi se pronašla najveća plata među svim zaposlenima u kompaniji. Skalarni podupit vraća tačno jednu vrednost - u ovom slučaju, najvišu platu.
3. **Izračunavanje razlike u plati**: Kolona **RazlikaUPlati** se izračunava oduzimanjem plate svakog zaposlenog od najviše plate u kompaniji, što je rezultat skalarnog podupita. Ova razlika pokazuje koliko svaki zaposleni zarađuje manje u poređenju sa najbolje plaćenim zaposlenim.

Ovaj upit pruža jasan uvid u razlike u platama unutar kompanije, identifikujući koliko svaki zaposleni zarađuje manje u odnosu na najvišu platu. Ova informacija može biti korisna za razmatranje strukture plata, potencijalne revizije platnih razreda ili za analizu platne politike kompanije.

Zadatak 58 – Podupit za izračuvanje broja dana

*Kako biste kreirali SQL upit koji koristi podupite kao izraze kolona za izračunavanje broja dana do najbliže narudžbine za svakog kupca, pretpostavljajući tabelu **Kupci** koja sadrži kolone za šifru kupca, ime i prezime, i tabelu **Narudzbine** koja sadrži kolone za šifru narudžbine, šifru kupca, i datum narudžbine, i kako biste osigurali da se u rezultatima prikažu ime i prezime kupca, kao i broj dana do njegove najbliže narudžbine u odnosu na trenutni datum?*

SQLgpt: Da bismo kreirali SQL upit koji koristi podupite kao izraze kolona za izračunavanje broja dana do najbliže narudžbine za svakog kupca, možemo koristiti **JOIN** operaciju između tabela

Kupci i **Narudžbine**, a zatim koristiti funkciju **DATEDIFF()** kako bismo izračunali broj dana između trenutnog datuma i datuma najbliže narudžbine. Evo kako bi taj upit mogao izgledati:

```
SELECT
    K.Ime,
    K.Prezime,
    DATEDIFF(day, GETDATE(), MIN(N.DatumNarudzbine))
      AS DaniDoNajblizeNarudzbine
FROM
    Kupci K
LEFT JOIN
    Narudzbine N ON K.ID_kupca = N.ID_kupca
GROUP BY
    K.Ime, K.Prezime;
```

Objašnjenje SQL upita:

1. **Selekcija informacija o kupcu**: Glavni deo upita selektuje ime i prezime svakog kupca iz tabele **Kupci**.
2. **Podupit kao izraz kolone**: Za svakog kupca, koristi se podupit unutar **SELECT** klauzule da se izračuna broj dana do njegove najbliže buduće narudžbine. Podupit koristi funkciju **DATEDIFF()** da izračuna razliku u danima između trenutnog datuma (**GETDATE()**) i datuma najbliže buduće narudžbine (**MIN(N.DatumNarudžbine)**).
3. **Filtriranje narudžbina**: Podupit filtrira narudžbine za svakog kupca (gde **N.ID_kupca = K.ID_kupca**) i uzima u obzir samo one narudžbine koje su zakazane za budućnost (**N.DatumNarudžbine > GETDATE()**), koristeći funkciju **MIN()** da pronađe najraniji datum narudžbine koji dolazi nakon trenutnog datuma.

Ovaj upit omogućava pregled kada se očekuje da će svaki kupac izvršiti svoju sledeću narudžbinu, što može biti korisno za planiranje zaliha, optimizaciju logistike ili za usmeravanje marketinških kampanja ka kupcima koji se očekuje da će uskoro izvršiti narudžbinu.

Zadatak 59 – Podupit kao izraz kolona

*Kako biste kreirali SQL upit koji koristi podupite kao izraze kolona za izračunavanje broja dana do najbliže narudžbine za svakog kupca, pretpostavljajući tabelu **Kupci** koja sadrži kolone za šifru kupca, ime i prezime, i tabelu **Narudzbine** koja sadrži kolone za šifru narudžbine, šifru kupca, i datum narudžbine, i kako biste osigurali da se u rezultatima prikažu ime i prezime kupca, kao i broj dana do njegove najbliže narudžbine u odnosu na trenutni datum?*

SQLgpt: Da bismo kreirali SQL upit koji koristi podupite kao izraze kolona za izračunavanje broja dana do najbliže narudžbine za svakog kupca, možemo koristiti podupit unutar **SELECT** klauzule. Evo kako bi taj upit mogao izgledati:

```
SELECT
    K.Ime,
    K.Prezime,
    (
        SELECT
            DATEDIFF(day, GETDATE(), MIN(N.DatumNarudzbine))
        FROM
```

```
            Narudžbine N
        WHERE
            N.ID_kupca = K.ID_kupca
            AND N.DatumNarudzbine > GETDATE()
    ) AS DaniDoNajblizeNarudzbine
FROM
    Kupci K;
```

Objašnjenje upita:

1. **Selekcija informacija o kupcu**: Glavni deo upita bira ime i prezime svakog kupca iz tabele **Kupci**.
2. **Podupit kao izraz kolone**: Za svakog kupca, koristi se podupit unutar **SELECT** klauzule da se izračuna broj dana do njegove najbliže buduće narudžbine. Podupit koristi funkciju **DATEDIFF()** da izračuna razliku u danima između trenutnog datuma (**GETDATE()**) i datuma najbliže buduće narudžbine (**MIN(N.DatumNarudžbine)**).
3. **Filtriranje narudžbina**: Podupit izdvaja narudžbine za svakog kupca (gde **N.ID_kupca = K.ID_kupca**) i uzima u obzir samo one narudžbine koje su zakazane za budućnost (**N.DatumNarudžbine > GETDATE()**), koristeći funkciju **MIN()** da pronađe najraniji datum narudžbine koji dolazi nakon trenutnog datuma.

Ovaj upit omogućava pregled kada se očekuje da će svaki kupac izvršiti svoju sledeću narudžbinu, što može biti korisno za planiranje zaliha, optimizaciju logistike ili za usmeravanje marketinških kampanja ka kupcima koji se očekuje da će uskoro izvršiti narudžbinu.

Zadatak 60 – Podupit kao filter za identifikaciju zaposlenih

*Kako biste kreirali SQL upit koji koristi podupite kao filtere za identifikaciju zaposlenih u tabeli **Zaposleni** koji imaju iskustvo rada na projektima iz određene oblasti, na primer, „IT”, pretpostavljajući tabelu **Zaposleni** koja sadrži kolone za šifru zaposlenog, ime i prezime, i tabelu **Projekti** koja sadrži kolone za šifru projekta, naziv projekta, oblast i šifru zaposlenog koji je radio na projektu, i kako biste osigurali da se u rezultatima prikažu samo relevantne informacije o zaposlenima koji imaju iskustvo u oblasti IT?*

SQLgpt: Da bismo kreirali SQL upit koji koristi podupite kao filtere za identifikaciju zaposlenih u tabeli **Zaposleni** koji imaju iskustvo rada na projektima iz određene oblasti, na primer, „IT”, možemo koristiti podupit unutar **WHERE** klauzule. Evo kako bi taj upit mogao izgledati:

```
SELECT
    Z.ID_zaposlenog,
    Z.Ime,
    Z.Prezime
FROM
    Zaposleni Z
WHERE
    EXISTS (
        SELECT
            1
        FROM
            Projekti P
        WHERE
            P.ID_zaposlenog = Z.ID_zaposlenog
```

```
            AND P.Oblast = 'IT'
    );
```

Objašnjenje upita:

1. **Selekcija informacija o zaposlenima**: Glavni deo upita izdvaja šifru zaposlenog, ime i prezime iz tabele **Zaposleni**.
2. **Korišćenje EXISTS kao filtera**: **EXISTS** klauzula se koristi u **WHERE** delu upita da proveri postojanje redova koji zadovoljavaju određeni uslov. U ovom slučaju, **EXISTS** proverava da li postoji bilo koji red u tabeli **Projekti** koji odgovara trenutno razmatranom zaposlenom (**P.ID_zaposlenog = Z.ID_zaposlenog**) i koji je povezan sa projektom u oblasti "IT" (**P.Oblast = 'IT'**).
3. **Filtriranje zaposlenih sa IT iskustvom**: Korišćenjem podupita unutar **EXISTS**, upit efikasno filtrira i uključuje samo one zaposlene koji imaju iskustvo rada na projektima u oblasti IT. Ako **EXISTS** klauzula pronađe barem jedan takav red, red iz spoljnog upita (tj. zaposleni) će biti uključen u rezultate.

Ovaj upit omogućava identifikaciju zaposlenih sa relevantnim iskustvom u IT projektima, što može biti korisno za potrebe regrutacije, formiranje timova za nove projekte ili za procenu veština i iskustva unutar organizacije.

Zadatak 61 – Generisanje sledeće vrednosti primarnog ključa

*Kako biste kreirali SQL upit koji generiše sledeću vrednost primarnog ključa za unos u tabelu **Narudzbine**, pretpostavljajući da tabela **Narudzbine** sadrži kolone za šifru narudzbine (koji je primarni ključ), datum narudžbine, šifru kupca, i ukupan iznos, i kako biste osigurali da se nova narudžbina unese bez rizika od dupliranja primarnog ključa, uzimajući u obzir da se šifra narudžbine automatski inkrementira za svaku novu narudžbinu?*

SQLgpt: Ako je šifra narudžbine automatski inkrementirani primarni ključ, nije potrebno eksplicitno generisati sledeću vrednost primarnog ključa prilikom unosa nove narudžbine. Umesto toga, možete jednostavno izostaviti kolonu sa šifrom narudžbine iz **INSERT** naredbe, a baza podataka će automatski dodeliti sledeću vrednost. Evo kako bi izgledala takva **INSERT** naredba:

```
INSERT INTO Narudzbine
   (DatumNarudzbine, ID_kupca, UkupanIznos)
VALUES
   (GETDATE(), @ID_kupca, @UkupanIznos);
```

Objašnjenje upita:

1. **Izostavljanje šifre narudžbine**: U **INSERT** naredbi, kolona šifre narudžbine nije navedena, jer baza podataka automatski dodeljuje vrednost ovom polju koristeći mehanizam automatske inkrementacije.
2. **Specifikacija ostalih kolona**: Navedene su kolone **DatumNarudžbine**, **ID_kupca**, i **UkupanIznos**, koje su potrebne za kreiranje nove narudžbine.
3. **Dodeljivanje vrednosti**: Vrednosti koje treba uneti (**GETDATE()** za trenutni datum, **@ID_kupca** i **@UkupanIznos**) su određene u **VALUES** klauzuli. Ovde se pretpostavlja da su **@ID_kupca** i **@UkupanIznos** promenljive ili parametri koji sadrže odgovarajuće vrednosti za novu narudžbinu.