Prijava na bazu podataka - SQLTools

Hostname: **???**
Username : **???**
Password: **???**
Host string: **etflab**

Kreiranje i manipulacija sa tabelama

Proces kiraranja tabele je više nego standardiziran, tako da je osnovna sintaksa za kreiranje tabele više nego ista kod svih dostupnih baza podataka:

```
SQL>  CREATE TABLE naziv_tabele
      ( kolona1 tip_podatka [not null],
        kolona2 tip_podatka [not null],
        kolona3 tip_podatka [not null],
        ...
      );
```

Imajući u vidu gore napisano, jednostavan primjer kreiranja tabele projekata izgledao bi kako slijedi:

```
SQL>  CREATE TABLE projekti(projekat_id       NUMBER,
                            naziv             VARCHAR2(100),
                            broj_zaposlenih   NUMBER,
                            status_projekta   CHAR(1),
                            datum_pocetka     DATE,
                            datum_kraja       DATE,
                            odgovorna_osoba   VARCHAR2(50));
```

Ovaj jednostavni izkaz prilikom izvršenja će kreirati tabelu projekata sa sedam kolona: projekat id, naziv, broj zaposlenih, status projekta, datum pocetka, datum kraja i odgovorna osoba. U kolonama naziv, status projekta i odgovorna osoba mogu se unijeti vrijednosti samo do određene dužine, tj. u koloni naziv može se unijet string do 100 karaktera, u koloni status samo jedan karakter i u koloni odgovorana osoba maksimalno 50 karaktera. Dakle sa definisanjem tipa podatka kolone uvode se i restrikcije po tipu i veličini podataka koji se mogu unijeti u krieirani tip polja.

Naziv tabele
Prilikom kreiranja tabele treba se pridržavati nekoliko jednostavnih pravila. Prva i osnova stvar je da naziv tabele ne može sadržavati više od 30 karaktera, niti može biti jedna od rezervisanih riječi baze podataka. Isto tako nekorektno je da prvi karakter u imenu tabele bude broj. Od simbola u nazivu tabele mogu se koristi karakteri kao što su: _, #, $, @, i ime tabele mora biti jedinstveno u okviru date šeme baze podataka u kojoj se tabela kreira.

Naziv kolone
Ista pravila koja važe prilikom određivanja i kreiranja tabele, važe i za nazive kolona. Međutim, za razliku od naziva tabele, imana kolona mogu se ponavljati u različitim tabelama u istoj šemi baze podataka. Jedino ograničenje je da naziv kolone u okviru iste tabele mora biti jedinstveno. Na primjer, dozvoljeno je da nazivi kolona department_id u tabelama employees i departments budu ista.

Tip podataka u koloni

Kao što je prirodno da prilikom programiranja u nekom od programskih jezika koristi tip podatka za svaku varijablu, isto ovo pravilo vrijedi i za kreiranje tabela. Drugim riječima mora se reći bazi podataka koji tip podatka može sačuvati u okviru definisane kolone. Tako na primjer, za kolone koje su definisane kao datumski tip podataka mogu se unijeti samo datumi. Baza podataka u ovom slučaju očekuje od korisnika da unese datumske vrijednosti u odgovarajuću kolonu.

U okviru Oracle baze podataka dozvoljeni su neki od sljedećih tipova podataka, prilikom kreiranja tabele:

Tip	Opis
VARCHAR2(broj karaktera)	Alfanumerički podatak varijabilne dužine. Prilikom deklaracije ovog tipa podatka potrebno je definisati maksimalano dozvoljeni broj karaktera. Minimalan broj karaktera za ovaj tip podatka je 1, a maksimalan 4000 karaktera.
CHAR(broj karaktera)	Alfanumerički podatak fiksne fiksne dužine izražen u bajtima. Minimalan broj karaktera za ovaj tip podatka je 1, a maksimalan 2000 karaktera.
NUMBER(p,s)	Brojevi mogu imati 2 argumenta. Prvi argument *p* označava preciznost broja, odnosno broj cifara koji je dozvojen u datom broju, dok drugi argument *s* označava broj decimalnih mjesta koji će se koristiti u ukviru deklarisane koloni sa ovim tipom podatka.
DATE	Tip podatka za datum i vrijeme
LONG	Alfanumerički stringovi veličine do 2 gigabytes,
RAW i LONG RAW	RAW binarni tip podatka veličine do 2 gigabytes.
BLOB	Binarni tip podataka veličine do 4 gigabytes
CLOB	Single-byte karakterni tip podatka veličine do 4 gigabytes.
BFILE	Binarni tip podatka za pohranjivanje u eksternom file-u, veličine do 4 gigabytes.

Kreiranje tabela upotrebom podupita

Jedna od čestih situacija u praksi je da se od postojeće ili postojećih tabela kreira nova tabela sa specifičnim brojem i tipom kolona vraćenih iz podupita. Sintaksa za kreiranje ovog tipa tabele data je sljedećom strukturom:

```
SQL>  CREATE TABLE  nazivTabele (
      ( kolona1,
        kolona2,
        kolona3,
        ...
      ) AS SELECT kolona1,
                  kolona2,
                  kolona3,
                  ...
            FROM tabela1, tabela2, ...
           WHERE tabela1.kolona1 = tabela2.kolona3
                   AND ...;
```

U sintaksi je:

nazivTabele - je naziv tabele
kolona 1..n - je naziv kolone, podrazumijevajuće vrijednosti i integriteti ograničenja

podupit - je upit koji definiše set slogova koji će biti insertovani u novu tabelu.

Prilikom izvršenja ovog iskaza kreirati će se nova tabela i insertovati podaci koji su vraćeni po izvršenju podupita. Na ovaj način se kreira nova tabela sa odgovarajućom strukturom i tipom podataka putem naznačenog podupita, a rezultati select iskaza kopiraju u novo formiranu tabelu.

Na primjer, pretpostavimo da je potrebno kreirati novu tabelu zaposlenih koja će sadržavati šifru zaposlenog, naziv zaposlenog, platu, šifru odjela, naziv odjela, šifru posla i naziv posla svih zaposlenih koji dobivaju dodatak na platu.

```
SQL> CREATE TABLE zaposleni(sifra_zaposlenog,
                            naziv_zaposlenog,
                            plata,
                            sifra_odjela,
                            naziv_odjela,
                            sifra_posla,
                            naziv_posla) AS
                    SELECT e.employee_id,
                            e.last_name || ' ' || e.first_name,
                            e.salary,
                            d.department_id,
                            d.department_name,
                            j.job_id,
                            j.job_title
                    FROM employees e, departments d, jobs j
                    WHERE e.department_id = d.department_id
                           AND e.job_id = j.job_id
                           AND e.commission_pct IS NOT NULL;
```

Može se vidjeti da se ovaj zadatak može riješiti i na sljedeći način:

```
SQL> CREATE TABLE zaposleni AS
                SELECT e.employee_id sifra_zaposlenog,
                        e.last_name || ' ' || e.first_name naziv_zaposlenog,
                        e.salary plata,
                        d.department_id sifra_odjela,
                        d.department_name naziv_odjela,
                        j.job_id sifra_posla,
                        j.job_title naziv_posla
                    FROM employees e, departments d, jobs j
                    WHERE e.department_id = d.department_id
                            AND e.job_id = j.job_id
                            AND e.commission_pct IS NOT NULL;
```

ALTER TABLE iskaz

Prilikom kreiranja šeme baze podataka i nakon kreiranja i puštanja u zvanični rad radne verzije šeme može se pojavati, a obično je tako, potreba za mnogim daljim promjenama u strukturi baze podataka, a samim tim i u strukturi definisanih tabela koje su kreirane nad datom šemom. ALTER TABLE je jedan od takvih iskaza koji omogućavaju administratoru baze podataka ili dizajneru šeme baze podataka da promijeni strukturu tabele nakon što je kreirana. ALTER TABLE komanda omogućava:

- Dodavanje nove kolone postojećoj tabeli.
- Promjena kolone koja već postoji kreirana u okviru tabele.
- Brisanje nepotrebne/ih kolone/a.
- Proglašavanje nepotrebnih kolona «neupotrebljivim».

Sintaksa ALTER TABLE iskaza je sljedeća:
- dodavanje nove kolone

```
SQL>  ALTER TABLE naziv_tabele ADD (naziv_kolone tip_podatka);
```

- modifikacija postojeće kolone

```
SQL>  ALTER TABLE naziv_tabele MODIFY (naziv_kolone tip_podatka);
```

- brisanje postojeće kolone

```
SQL>  ALTER TABLE naziv_tabele DROP COLUMN (naziv_kolone);
```

- proglašavanje nepotrebne kolone neupotrebljivom

```
SQL>  ALTER TABLE naziv_tabele SET UNUSED (naziv_kolone);
```

Brisanje nepotrebnih kolona koje su označene kao neupotrebljive vrši se sa sljedećom komandom:

```
SQL>  ALTER TABLE naziv_tabele DROP UNUSED COLUMNS;
```

Komanda za brisanje kompletne tabele sa strukturom i svim podacima je:

```
SQL>  DROP TABLE naziv_tabele;
```

Komanda za promjenu imena tabele:

```
SQL>  RENAME staro_ime_tabele to novo_ime_tabele;
```

Komentari za tabele i kolone

Jedna od osobina dobrog dizajna šeme baze podataka je upotreba komentara za tabele i kolone. U poslovnim sistemima gdje na jednom projektu radi više osoba na dizajnu i implementaciji, neophodno je da sve osobe poznaju sistem što je moguće više i detaljnije, tj. do najsitnijih detalja ako je ikako moguće. Iz tih praktičnih razloga jako je važno 'znati' šta koja od tabela i kolona označava i na šta se odnosi u sistemu. Naima, poznavanje sistema je od suštinske važnosti prilikom projektovanja i implementacije sistema, iz tih raloga jako je praktično da se u svakom momentu u produkcionom sistemu može znati šta koja od tabela i kolona znači. To se radi tako što se uvode komentari za tabele i kolone. Sintaksa za Oracle bazu podataka za komentare data je generalno sljedećom sintaksom:

```
SQL>  COMMENT ON TABLE tabela / COLUMN tabela.kolona / IS 'komentar';
```

U sintaksi je:

tabela	- je naziv tabele za koju se dodaje komentar
table.kolona	- je naziv kolone u tabeli za koju se dodaje komentar
komentar	- je tekst komentara koji se dodaje odgovarajućoj tabeli ili koloni odgovarajuće tabele.

Na primjer, neka je potrebno dodati komentar za tabelu zaposlenih «Tabela svih zaposlenih u korporaciji XYZ», i za kolonu employee_id komentar «Kolona koja jednoznačno određuje sve ostale atribute tabele».

```
SQL> COMMENT ON TABLE employees IS 'Tabela svih zapolenih u korporaciji XYZ';

SQL> COMMENT ON COLUMN employees.employee_id IS 'Kolona koja jednoznačno određuje sve ostale atribute tabele ';
```

Komentari se pohranjuju u data dictionary i mogu se vidjeti kroz jedan od sljedećih pogleda putem data dictionary-a u COMMENTS kolonama:

- ALL_COL_COMMENTS
- USER_COL_COMMENTS
- ALL_TAB_COMMENTS
- USER_TAB_COMMENTS

Sve kreirane tabele u okviru konektovanog korisnika mogu se vidjeti kroz jednu od sljedećih tabela data dictionary-a:

- USER_TABLES
- USER_OBJECTS
- USER_CATALOG

NULL vrijednost
Kada se kreira tabela daje se mogućnost programeru ili administratoru baze podataka da kreira tabelu sa kolonama u koje je dozvoljeno unositi ili ne NULL vrijednosti. Za kolone koje ne smiju sadržavati NULL vrijednosti potrebno je prilikom kreiranje tabele u SQL označiti da kolona ne smije sadržavati NULL vrijednost sa ključnim riječima NOT NULL. NOT NULL označava da svaki slog mora sadržavati vrijednost u naznačenoj koloni.

Na primjer, neka je potrebno kreirati novu tabelu informacija sa kolonom info koja ne smije sadržavati NULL vrijednost:

```
SQL> CREATE TABLE informacije(info VARCHAR2(100) NOT NULL);
```

Zadaci

1. Kreirati tabelu odjela sa sljedećom strukturom:

Kolona	Tip	Dužina	Not null
Id	Varchar2	25	Da
Naziv	Varchar2	10	Da
Opis	Char	15	Ne
Datum	Date	-	Da
Korisnik	Varchar2	30	Da
Napomena	Varchar2	10	Ne

2. Kopirate podatke iz tabele odjela u vašu kreiranu tabelu odjela samo onim podacima koje je moguće upisati u tabelu.
3. Modificirati vašu tabelu odjela na takav način da se upišu svi podaci iz postojeće tabele odjela.
4. Kreiraje vašu tabelu zaposlenih sa sljedećom strukturom:

Kolona	Tip	Dužina	Not null
Id	Number	4	Da
Sifra_zaposlenog	Varchar2	5	Da
Naziv_zaposlenog	Char	50	Ne
Godina_zaposlenja	Number	4	Da
Mjesec_zaposlenja	Char	2	Da
Sifra_odjela	Varchar2	5	Ne
Naziv_odjela	Varchar2	15	Da
Grad	Char	10	Da
Sifra_posla	Varchar2	25	Ne
Naziv_posla	Char	50	Da
Iznos_dodatak_na_platu	Number	5	Ne
Plata	Number	6	Da
Kontinent	Varchar2	20	Ne
Datum_unosa	Date	-	Da
Korisnik_unio	Char	20	Da

5. Na sve postojeće podatke o zaposlenim iz tabla vezanih za zaposlene, kopirajte potrebne podatke u vašu tabelu zaposlenih.
6. Kopirajte vašu tabelu zaposlenih u novu tabelu zaposleni2.
7. Modificirajte tabelu zaposleni2 tako da kolone: sifra_zaposlenog i naziv zaposlenog, sifra_odjela i naziv_odjela, sifra_posla i naziv_posla bude predstavljen kao jedna kolona respektivno: odjel, zaposleni i posao.
8. Promijeniti naziv tabele zaposleni2 u naziv zap_backup.
9. Dodajte komentare za vaše kreirane tabele odjela i zaposlenih, koji precizno označavaju šta koja od tabela znači i sadržava.
10. Dodajte komentare za vaše kreirane kolone tabela odjela i zaposlenih, koji precizno označavaju šta koja od kolona znači i sadržava.
11. Proglasite kolone datum_unosa i korisnik_unio u vašoj zap_backup tabeli neupotrebljivim.
12. Izlistati sve opisane komentare za vaše tabele odjela i zaposlenih.
13. Izbrisati sve neupotrebljive kolone u vašoj tabeli zaposlenih.