Database Programming with SQL
kurs 2017 – database design and programming with sql
students slajdovi

4-1 Case and Character Manipulation

- Pri radu sa podacima iz db, način na koji se oni prikazuju je važna detalj
- Najčešće u SQL, treba promeniti način na koji se podaci pokazuju u zavisnosti od zahteva zadatka koji treba da rešimo

Dual Table

- Tabela DUAL ima jedan red nazvan X i jednu kolonu nazvanu DUMMY
- DUAL tabela se koristi za kreiranje SELECT iskaza i izvršenje funkcija koje nisu direktno u relaciji sa specifičnom tabelom iz db
- Upiti koji koriste DUAL tabele vraćaju jedan red kao rezultat; DUAL može biti korisna za izradu kalkulacija i za evaluaciju izraza koji nisu izvučeni iz tabele
- Dual tabela se koristi za učenje single-row funkcija
- U ovom primeru DUAL tabela se koristi za izvršenje SELECT iskaza koji sadrži kalkulacije
- Vidi se da SELECT vraća vrednost koja ne postoji u DUAL tabeli, vraćena vrednost je rezultat izvršenih kalkulacija

```
SELECT (319/29) + 12
FROM DUAL;
```

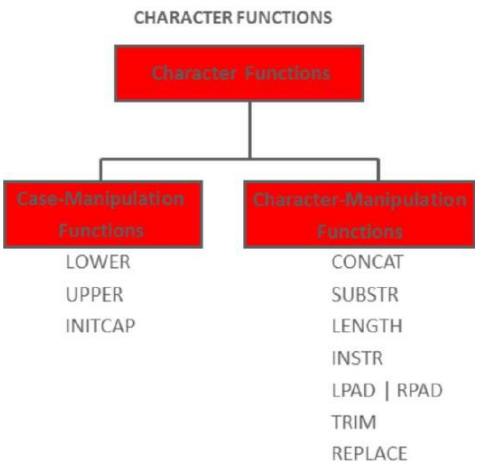
(319/29)+12
23

Single-Row Character Functions

- Single-row karakter funkcije su podeljene u dve kategorije: funkcije koje konvertuju veličinu stringova karaktera, funkcije koje mogu join, extract, show, find, pad i trim string karaktere
- Single-row funkcije se mogu koristiti u SELECT, WHERE i ORDER BY rečenice
- Single-row funkcije su veoma snažni predefinisani kod koji prihvata argumente i vraća vrednost; argument može biti definisan kao ime kolone, izraz ili konstanta
- Funkcije za manipulaciju sa veličinom slova su važne pošto ne možete uvek znati u kojoj velični slova (case) je podatak smešten u db
- Case manipulacija omogućava privremenu konverziju db podatka u željenu veličinu
- Nepodudarnost između db case storage i upita case request je izbegnuto

Case-Manipulation Functions

- Case-manipulation funkcije se koriste za konvertovanje podataka iz stanja u kojem su smeštene u tabeli u lower, upper ili mixed case
- Ove konverzije se mogu koristiti za formatiranje izlaza i takođe koristiti za pretragu posebnih stringova
- Case-manipulation funkcije se mogu koristiti u najvećem broju SQL iskaza
- Case-manipulation funkcije su često pomoć kada se traži podatak ali se ne zna da li podatak koji se traži je u upper ili lower case



- Sa stanovišta db, 'V' i 'v' nisu isti karakteri i kao takvi treba potražiti korišćenje pravilnog case
- LOWER(column|expression) konvertuje alfa karaktere u lower-case

```
SELECT last_name
FROM employees
WHERE LOWER(last_name) = 'abel';
```

- Razlog zašto Oracle razlikuje 'V' i 'v' je zbog načina na koji se smeštaju karakteri; on nesmešta znakove direktno, nego njihove odgovarajuće binarne vrednosti, u zavisnosti od seta karaktera u db
- U većem delu zapadnog sveta, ASCII set karaktera će se koristiti kao db set karaktera i binarni kod za 'V' i 'v' su različiti brojevi, zato Oracle ne smatra njih istima
- UPPER(column|expression) konvertuje alfa karaktere u upper-case

```
SELECT last_name
FROM employees
WHERE UPPER(last_name) = 'ABEL';
```

- INITCAP(column|expression) konvertuje alfa karakter vrednosti u uppercase za prvo slovo svake reči

```
SELECT last_name
FROM employees
WHERE INITCAP(last_name) = 'Abel';
```

- Korišćenje Case manipulacije funkcije unutar WHERE rečenice omogućava povratak redova bez obzira na case kako su smeštene u tabeli
- Korišćenje funkcije case manipulacije u SELECT rečenici menja način na koji su prikazani rezultati upita

Character Manipulation Functions

- Funkcije karakter manipulacije se koriste za ekstrakciju, izmenu, formatiranje ili izmenu karakter stringa na neki način
- Jedan ili više karaktera ili reči se passed u funkciju a funkcija će onda izvesti svoju funkcionalnost na ulaznom stringu karaktera i vratiti izmenjenu, proširenu, izbrojanu ili promenjenu vrednost
- CONCAT: Joins (udružuje) dve vrednosti zajedno
- Uzima dva string karakter argumenta i udružuje drugi string na prvi; takođe se može napisati korišćenjem concatenation operatora: 'Hello' || 'World'

Examples:	Result
`SELECT CONCAT('Hello', 'World')` `FROM DUAL;`	HelloWorld
`SELECT CONCAT(first_name, last_name)` `FROM employees;`	EllenAbel CurtisDavies ...

- SUBSTR: Extracts string određene dužine
- Argumenti su (character String, starting position, length)
- Argument Length je opcion, i ako je izbegnut, vraća sve karaktere na kraj stringa

Examples:	Result
`SELECT SUBSTR('HelloWorld', 1, 5) FROM DUAL;`	Hello
`SELECT SUBSTR('HelloWorld', 6) FROM DUAL;`	World
`SELECT SUBSTR(last_name, 1, 3) FROM employees;`	Abe Dav

- Primer 1: ekstrakuje substring od 5 karaktera iz pozicije 1 od 'HelloWorld'
- Primer 2: ekstrakuje substring počevši od pozicije 6 'HelloWorld' na kraj stringa
- Primer 3: ekstrakuje substring prva 3 karaktera iz employee prezimena
- LENGTH: Pokazuje dužinu stringa kao brojčanu vrednost; funkcija uzima string karaktera kao argument a vraća broj karaktera u tom stringu

Examples:	Result
`SELECT LENGTH('HelloWorld') FROM DUAL;`	10
`SELECT LENGTH(last_name) FROM employees;`	4 6 ...

- INSTR: Nalazi numeričku poziciju specificiranog karaktera; traži prvo pojavljivanje substringa unutar stringa karaktera a vraća poziciju kao broj; ako se substring ne nađe, vraća se broj nula

Examples:	Result
`SELECT INSTR('HelloWorld', 'W') FROM DUAL;`	6
`SELECT last_name, INSTR(last_name, 'a') FROM employees;`	Abel 0 Davies 2 ...

- Primer 1: 'W' je šesti karater iz argumenta prvog stringa, tako da će funkcija vratiti 6
- Primer 2: vraća prvo pojavljivanje karaktera 'a' u prezimenu zaposlenih; ako ime ne sadrži 'a', vraća se nula; čak iako Abel sadrži 'A', nije dobra veličina i zato se 0 vraća
- LPAD: Popunjava sa leve strane string karaktera, šro rezultuje vrednosti koja je poravnata sa desne strane (right-justified); traži tri argumenta: string karaktera, ukupan broj karaktera u popunjenom stringu, karakter sa kojim će se popuniti višak mesta u stringu

Examples:	Result
`SELECT LPAD('HelloWorld', 15, '-') FROM DUAL;`	-----HelloWorld
`SELECT LPAD(last_name, 10, '*') FROM employees;`	******Abel ****Davies ...

- Primer 1: string 'HelloWorld' je dopunjen sa leve strane za 15 mesta simbolom –
- Primer 2: prezime zaposlenih je dopunjeno sa leve strane do 10 mesta sa *
- RPAD: Dopunjava sa desne strane string karaktera, praveći left-justified vrednost

Examples:	Result
`SELECT RPAD('HelloWorld', 15, '-') FROM DUAL;`	HelloWorld-----
`SELECT RPAD(last_name, 10, '*') FROM employees;`	Abel****** Davies**** ...

- TRIM: Uklanja sve specified karaktere sa početka ili kraja ili sa obe strane stringa

Examples:	Result
`SELECT TRIM(LEADING 'a' FROM 'abcba') FROM DUAL;`	bcba
`SELECT TRIM(TRAILING 'a' FROM 'abcba') FROM DUAL;`	abcb
`SELECT TRIM(BOTH 'a' FROM 'abcba') FROM DUAL;`	bcb

- Primer 1: uklanja početno a sa početka stringa
- Primer 2: uklanja poslednje a iz stringa
- Primer 3: uklanja prvo i poslednje a iz stringa
- Ako nisu napisane LEADING, TRAILING ili BOTH, funkcija vraća BOTH
- Ako specificirani karakter nije prvi ili poslednji u stringu, neće doći do uklanjanja karaktera
- REPLACE: zamenjuje sekvencu karaktera u stringu sa drugim setom karaktera

```
REPLACE (string1, string_to_replace, [replacement_string] )
```

- String1 je string koji ima karaktere koji će biti zamenjeni; string_to_replace je string koji će se tražiti unutar stringa1 za zamenu; replacement_string je novi string koji će biti unet kao zamena u string1

Examples:	Result
`SELECT REPLACE('JACK and JUE', 'J', 'BL') FROM DUAL;`	BLACK and BLUE
`SELECT REPLACE('JACK and JUE', 'J') FROM DUAL;`	ACK and UE
`SELECT REPLACE(last_name, 'a', '*') FROM employees;`	Abel D*vies De H**n

- Primer 1: Svako pojavljivanje od J u stringu 'JACK and JUE' je zamenjena sa BL
- Primer 2: Ako nije postavljen replacement_string, string_to_replace se briše iz stringa
- Primer3: Svako pojavljivanje a u prezimenu zaposlenog je zamenjeno sa *

Using Column Aliases With Functions

- Sve funkcije rade sa vrednostima koje su u zagradama a svaka funkcija denotes svoju svrhu što je korisno zapamtiti pri pravljenju upita; često skraćenice kolona se koriste za imenovanje funkcija; kada se koristi alijas kolone, on se pojavljuje na izlazu umesto prave sintakse funkcije

- U sledećim primerima, alijas “User Name“ je zamenjen sa sintaksom funkcije u prvom upitu
- Po difoltu, ime kolone u SELECT iskazu se pojavljuje kao naslov kolone
- U drugom primeru, ipak ne postoji kolona u tabeli za dobijeni rezultat pa se sintaksa upita umesto toga koristi

```
SELECT LOWER(last_name)|| LOWER(SUBSTR(first_name,1,1))
  AS "User Name"
FROM employees;
```

User Name
abele
daviesc
de haanl

```
SELECT LOWER (last_name)||LOWER(SUBSTR(first_name,1,1))
FROM f_staffs;
```

LOWER(LAST_NAME)||LOWER(SUBSTR(FIRST_NAME,1,1))
does
millerb
tuttlem

- Prvi primer koristi alias za naslov kolone i mnogo je čitljiviji i bolji za korisnika nego drugi primer koji nema alijas za kolonu

Substitution Variables

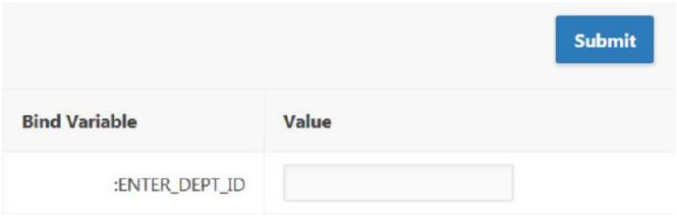
- Povremeno može zatrebati startovati isti upit sa mnogim različitim vrednostima da bi se dobio različit set rezultata
- Zamisli npr, ako moraš napisati izveštaj o zaposlenima i njihovim sektorima, ali upit mora vratiti samo podatke za jedan po jedan sektor
- Bez mogućnosti korišćenja promenjivih za zamenu, ovaj zahtev bi značio ponovno editovanje istog iskaza za promenu WHERE rečenice
- Srećom, APEX podržava promenjive za zamenu; da bi se one koristile treba samo zameniti vrednost u iskazu sa :named_variable
- APEX će onda pitati za vrednost pri izvršenju iskaza
- Ako je originalni upit:

```
SELECT first_name, last_name, salary, department_id
FROM employees
WHERE department_id= 10;
```

- Ako treba ga startovati i sa različitim vrednostima 20, 30, 40...može se napisati kao:

```
SELECT first_name, last_name, salary, department_id
FROM employees
WHERE department_id=:enter_dept_id;
```

- Pomoću : se prepoznaje tekst koji sledi kao da je promenjiva (:enter_dept_id)
- Promenjiva za zamenu je korisnički definisana u vreme izvršenja
- Kada se klikne na Run, pojavljuje se prozor u APEX:



- Da bi ovo funkcionisalo mora biti onemogućeni Pop-Up blockers, inače APEX ne može tražiti za vrednost promenjive
- Promenjive za zamenu se tretiraju kao string karaktera u APEX, što znači da pri davanju vrednosti karaktera ili datuma, ne treba koristiti apostrofe
- Tako da će WHERE rečenica izgleda ovako:

```
SELECT *
FROM  employees
WHERE last_name = :l_name;
```

4-2 Number functions

Number Functions

- Postoje tri brojčane funkcije: ROUND, TRUNC, MOD

ROUND

- ROUND se koristi i za brojeve i za datume; najviše se koristi za zaokruživanje brojeva na određeni broj decimalnih mesta ali takođe može zaokružiti brojeve levo od decimalnog zareza
- Sintaksa: ROUND(column|expression, decimal places)
- Primetiti da broj decimalnih mesta nije specificiran ili je nula, onda će broj biti zaokružen bez decimalnih mesta
- ROUND(45.926) 46
- ROUND(45.926, 0) 46
- Ako je broj decimalnih mesta pozitivan broj, broj je zaokružen na taj broj decimalnih mesta desno od decimalnog zareza
- ROUND(45.926, 2) 45.93
- Ako je broj decimalnih mesta negativan, broj je zaokružen na taj broj decimalnih mesta levo od decimalnog zareza
- ROUND(45.926, -1) 50

TRUNC

- Funkcija TRUNC se može koristiti i sa brojevima i sa datumima, Uglavnom se koristi za presecanje kolone, izraza ili vrednosti na specificirani broj decimalnih mesta
- Kada se TRUNC koristi, ako nije specificiran broj decimalnih mesta, onda poput ROUND, specificirani broj je po difoltu na nula
- Sintaksa: TRUNC(column|expression, decimal places)
- TRUNC(45,926, 2) 45.92
- Kao i sa ROUND, ako TRUNC izraz nije specificirao broj decimalnih mesta ili specificirao nulu, broj je odsečen bez decimalnih mesta
- TRUNC (45.926, 0) 45
- TRUNC (45.926) 45

- Znači da TRUNC ne zaokružuje brojeve već samo ih preseca na određenom mestu

MOD

- MOD funkcija nalazi ostatak deljenja (npr, MOD od 5 podeljeno sa 2 je 1)
- MOD se može koristiti za određivanje da li je vrdnost neparna ili parna. Ako se podeli vrednost sa 2 i nema ostatka, broj mora biti paran broj
- Npr, ako je MOD od x podeljen sa 2 daje 0, onda x mora biti parno
- Kolona “Mod Demo“ pokazuje da li je broj aerodroma svake zemlje paran ili neparan

```
SELECT country_name, MOD(airports,2)
 AS "Mod Demo"
FROM wf_countries;
```

- 1 means the number is odd, and even.

COUNTRY_NAME	Mod Demo
Canada	1
Republic of Costa Rica	0
Republic of Cape Verde	1
Greenland	0
Dominican Republic	0
State of Eritrea	1
...	